<목차>

2021년 5월 29일 토요일 오후 5:12

044 데이터 분석 함수로 순위의 비율 출력하기(CUME_DIST)
045 데이터 분석 함수로 데이터를 가로로 출력하기(LISTAGG)
046 데이터 분석 함수로 바로 전 행과 다음 행 출력하기(LAG, LEAD)
047 COLUMN을 ROW로 출력하기 1(SUM+DECODE)
048 COLUMN을 ROW로 출력하기 2(PIVOT)
049 ROW를 COLUMN으로 출력하기(UNPIVOT)
050 데이터 분석 함수로 누적 데이터 출력하기(SUM OVER)
051 데이터 분석 함수로 비율 출력하기(RATIO_TO_REPORT)
052 데이터 분석 함수로 집계 결과 출력하기 1(ROLLUP)
053 데이터 분석 함수로 집계 결과 출력하기 2(CUBE)
054 데이터 분석 함수로 집계 결과 출력하기 3(GROUPING SETS)
055 데이터 분석 함수로 출력 결과 넘버링 하기(ROW_NUMBER)

PART 3「중급」 SQL 실력 다지기
056 출력되는 행 제한하기 1(ROWNUM)
057 출력되는 행 제한하기 2(Simple TOP-n Queries)
058 여러 테이블의 데이터를 조인해서 출력하기 1(EQUI JOIN)
059 여러 테이블의 데이터를 조인해서 출력하기 2(NON EQUI JOIN)
060 여러 테이블의 데이터를 조인해서 출력하기 3(OUTER JOIN)
061 여러 테이블의 데이터를 조인해서 출력하기 4(SELF JOIN)
062 여러 테이블의 데이터를 조인해서 출력하기 5(ON절)
063 여러 테이블의 데이터를 조인해서 출력하기 5(USING절)
064 여러 테이블의 데이터를 조인해서 출력하기 6(NATURAL JOIN)
065 여러 테이블의 데이터를 조인해서 출력하기 7(LEFT/RIGHT OUTER JOIN)
066 여러 테이블의 데이터를 조인해서 출력하기 8(FULL OUTER JOIN)
067 집합 연산자로 데이터를 위아래로 연결하기 1(UNION ALL)
068 집합 연산자로 데이터를 위아래로 연결하기 2(UNION)
069 집합 연산자로 데이터의 교집합을 출력하기(INTERSECT)
070 집합 연산자로 데이터의 차이를 출력하기(MINUS)
071 서브 쿼리 사용하기 1(단일행 서브쿼리)
072 서브 쿼리 사용하기 2(다중 행 서브쿼리)
073 서브 쿼리 사용하기 3(NOT IN)
074 서브 쿼리 사용하기 4(EXISTS와 NOT EXISTS)
075 서브 쿼리 사용하기 5(HAVING절의 서브 쿼리)
076 서브 쿼리 사용하기 6(FROM절의 서브 쿼리)
077 서브 쿼리 사용하기 7(SELECT절의 서브 쿼리)
078 데이터 입력하기(INSERT)
079 데이터 수정하기(UPDATE)
080 데이터 삭제하기(DELETE, TRUNCATE, DROP)
081 데이터 저장 및 취소하기(COMMIT, ROLLBACK)
082 데이터 입력, 수정, 삭제 한번에 하기(MERGE)
083 락(LOCK) 이해하기
084 SELECT FOR UPDATE절 이해하기
085 서브 쿼리를 사용하여 데이터 입력하기
086 서브 쿼리를 사용하여 데이터 수정하기
087 서브 쿼리를 사용하여 데이터 삭제하기
088 서브 쿼리를 사용하여 데이터 합치기
089 계층형 질의문으로 서열을 주고 데이터 출력하기 1
090 계층형 질의문으로 서열을 주고 데이터 출력하기 2
091 계층형 질의문으로 서열을 주고 데이터 출력하기 3
092 계층형 질의문으로 서열을 주고 데이터 출력하기 4
093 일반 테이블 생성하기(CREATE TABLE)
094 임시 테이블 생성하기(CREATE TEMPORAY TABLE)
095 복잡한 쿼리를 단순하게 하기 1(VIEW)

096 복잡한 쿼리를 단순하게 하기 2(VIEW)
097 데이터 검색 속도를 높이기(INDEX)
098 절대로 중복되지 않는 번호 만들기(SEQUENE)
099 실수로 지운 데이터 복구하기 1(FLASHBACK QUERY)
100 실수로 지운 데이터 복구하기 2(FLASHBACK TABLE)
101 실수로 지운 데이터 복구하기 3(FLASHBACK DROP)
102 실수로 지운 데이터 복구하기 4(FLASHBACK VERSION QUERY)
103 실수로 지운 데이터 복구하기 5(FLASHBACK TRANSACTION QUERY)
104 데이터의 품질 높이기 1(PRIMARY KEY)
105 데이터의 품질 높이기 2(UNIQUE)
106 데이터의 품질 높이기 3(NOT NULL)
107 데이터의 품질 높이기 4(CHECK)
108 데이터의 품질 높이기 5(FOREIGN KEY)
109 WITH절 사용하기 1(WITH ~ AS)
110 WITH절 사용하기 2(SUBQUERY FACTORING)
111 SQL로 알고리즘 문제 풀기 1(구구단 2단 출력)
112 SQL로 알고리즘 문제 풀기 2(구구단 1단 ~ 9단 출력)
113 SQL로 알고리즘 문제 풀기 3(직각삼각형 출력)
114 SQL로 알고리즘 문제 풀기 4(삼각형 출력)
115 SQL로 알고리즘 문제 풀기 5(마름모 출력)
116 SQL로 알고리즘 문제 풀기 6(사각형 출력)
117 SQL로 알고리즘 문제 풀기 7(1부터 10까지 숫자의 합)
118 SQL로 알고리즘 문제 풀기 8(1부터 10까지 숫자의 곱)
119 SQL로 알고리즘 문제 풀기 9(1부터 10까지 짝수만 출력)
120 SQL로 알고리즘 문제 풀기 10(1부터 10까지 소수만 출력)
121 SQL로 알고리즘 문제 풀기 11(최대 공약수)
122 SQL로 알고리즘 문제 풀기 12(최소 공배수)
123 SQL로 알고리즘 문제 풀기 13(피타고라스의 정리)
124 SQL로 알고리즘 문제 풀기 14(몬테카를로 알고리즘)
125 SQL로 알고리즘 문제 풀기 15(오일러 상수 자연상수 구하기)

10/20 (001)

2020년 10월 20일 화요일 오후 2:55

- SQL이 무엇인지?
- SQL을 왜 배워야 하는지
- SQL 종류
- 기본 Select 문장

SQL의 종류

1. Query 문 ---> 데이터를 검색하는 언어
   SELECT문의 6가지 절

2. DML 문 ---> Data Manipulation Language (조작)
   A. Insert : 데이터 입력언어
   B. Update : 데이터 수정언어
   C. Delete : 데이터 삭제언어
   D. Merge: 데이터 입력, 수정, 삭제를 한번에 수행하는 명령어

3. DDL문 : Data Definition Language
   Create, alter, drop. Truncate. Rename
   생성 수정 삭제 삭제 이름변경

4. DCL문 : Data Control Language
   A. Grant :
      데이터를 접근할 수 있는 권한 부여
      Revoke :
   B. 데이터를 접근할 수 있는 권한을 취소

5. TCL문 : Transaction Control Language
   a. Commit : 현재의 데이터의 상태를 db에 영구히 저장하겠다.
   b. Rollback : 지금까지 했던 작업들을 다 취소하겠다.
   c. Savepoint : 특점시점까지 롤백하는 기능

Orcale Database 19C?
C = cloud의 약자
인공지능의 한 부분인 머신러닝 기능이 내장되어 있다.
머신러닝의 기능 : 컴퓨터가 스스로 데이터를 학습하여 공부하는 기능

Ex . 유방암 데이터 (양성, 악성 구분) -> 머신러닝 학습시켜 악성인지 양성인지 구분
사진을 보여주는 기법

기본 Select문장 연습

1. 도스창 열고 아래와 같이 타이핑, 접속

sqlplus "/ as sysdba"

2. SQL > show user 입력

SQL > "SYS"입니다

설명 : sys는 오라클의 최고 권한자이다.

3. 카페에서 SQL 명령어 찾아 복붙 , enter
4. Select ename, sal from emp;
   적어서 실행되는지 확인

   Select : 검색해라, 조회해라

   Ename : 이름

   Sal : 월급

   Emp : emp라는 테이블로부터

   ; (세미콜론) : 앞에 있는 문장 실행해라

❖ **Emp (사원) 테이블 컬럼 소개**

- Empno : 사원번호
- Ename : 사원이름
- Sal : 월급
- Job : 직업
- Comm : 커미션
- Mgr : 관리자 번호
- Hiredate : 입사일
- Deptno : 부서번호

10/21 (002-021)

2020년 10월 21일 수요일 오전 9:40

<복습>

- **SQL이 무엇인지?**
  구조적 질의 언어 - > database에서 데이터를 검색하고 조작하는 언어

  Data base의 종류 : 오라클, mssql, mysql, postgreSQL, Maria db

  - 약 80% - 오라클/ 게임회사나 소셜커머스 - mssql많이 씀 (빨리 지울 수 있어서)
  - 중요한 데이터는 oracle, 상대적으로 중요하지 않은 데이터 -> mysql, postgreSQL, Maria db (ex . 아프리카tv 채팅 대화 내용)

- **SQL을 왜 배워야 하는지**
  원하는 정보를 얻기 위해서, 고객의 니즈 파악
  - 물류회사 같은 경우 재고 현황 파악
  - 게임회사면 게임의 흥행 여부에 대한 원인 파악
  - 단순 검색 뿐 아니라 의미있는 정보를 얻어내는 검색

- **SQL 종류**
  1. Query
  2. DML문
  3. DDL문
  4. DCL문
  5. TCL문

- **기본 Select 문장**

  001 테이블에서 특정 열 (column)선택하기

  select ename, sal --------->컬럼
  from emp;

  select : 선택해라, 조회해라
  from : ~로 부터
  emp: 테이블 이름
  ; : 앞에 있는 문장을 실행해라

- ---

❖ **SQL 주의사항**

1. SQL은 대소문자를 구분하지 않는다.

   Ex) SELECT ENAME, SAL FROM EMP; ----->가능
   select ename, sal from emp; -----> 가능

2. SQL은 한줄로 작성하지 말고
   절 다음 라인으로 분리해서 작성한다.
   (점점 길어지기 때문)

   Ex) SELECT절
   FROM절

3. 들여쓰기를 사용해서 SQL의 가독성을 높인다.
   (+ 가급적 소문자로 작성하는 것이 오류찾기에 좋다)

   Ex) Select ename, sal
   From emp
   Where
   Group by
   Having
   Order by

질문:
Edit 어떻게 하는지
Ex) 컬럼 이름을 잘못 넣었을 때

답변:
명령크롬프트에 ed 치고 엔터치기
그러면 창이 하나 뜬다.

거기서 고치고 싶은 부분 고치고, 저장하고
끄기

그다음 / 누르고 enter 치면 된다.

- SQL developer 설치

- 이 툴은 도스창을 이용하지 않고 오라클에 접속해서 메모장 같은 화면에서 편하게 SQL을 작성할 수 있게 하는 툴

- 오라클에 접속하기 위해서는 접속 정보를 알아야 합니다.

1. 데이터 베이스 서버의 아이피 주소 (건물 주소) : 127.0.0.1
2. 포트번호(건물 내의 통로 번호) : 1522
3. 오라클 인스턴트 이름 (SID) (건물 내 특정 회사이름) : orcl2

위의 정보를 알아내는 명령어:
건물 내 경비원 아저씨에게 물어보면 된다.
---> 도스창을 열고 lsnrctl status라고 하고 엔터를 칩니다.

<명령어>

- 결과 화면의 가로 사이즈 조절하는 명령어
  Set lines 300 (SQLPLUS라는 툴의 명령어이지, SQL이 아님)

- 방금 수행했던 SQL다시 실행하기:
  / 하고 ENTER

- 결과 화면에서 세로 사이즈 조절하는 명령어
  Set pages 400

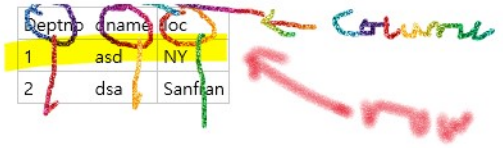
❖ 가로, 세로 사이즈 조절은 해당 창에서만 유효하고 다른 창을 열면 새롭게 다시 명령해 주어야 함.

002 테이블에서 모든 열(COLUMN) 출력하기

- Select *
  From emp;
  설명 : * 을 asterisk 라고 하고 모든 컬럼을 다 선택할 때 사용

- 테이블은 column과 row로 구성

Deptno	dname	loc
1	asd	NY
2	dsa	Sanfran



003 컬럼 별칭을 사용하여 출력되는 컬럼명 변경하기

* 컬럼별칭은 컬럼명 대신에 다른 컬럼명을 지정할 때 사용하는 문법입니다.

Ex) select ename as 이름 sal as 월급
From emp;

❖ 컬럼명 뒤에 as를 쓰고 그리고 컬럼별칭을 작성한다.
결과가 출력될 때 컬럼별칭이 출력된다.
As는 생략가능합니다.

004 연결 연산자 사용하기 (||)

" 연결 연산자는 두 컬럼의 데이터를 연결해서 출력하는 연산자 입니다."
|| ----> 엔터키 옆 원화표시 있는 것을 쉬프트 해서 두번 누르기

Ex) select ename || sal
From emp;

Ex) select ename || '의 월급은' || sal
From emp

------> 연결해서 값이 나옴

❖ 고치고 싶은 부분이 나왔을때, 해당 부분을 복사 붙여넣기 하고
edit 해당 문장
From emp (; 생략)
/
저장하기!!

005 중복된 데이터를 제거해서 출력하기(DISTINCT)

"dintinct 키워드를 컬럼명 앞에 작성하고 실행하면 중복된 데이터를 제거하고 출력할 수 있습니다."

Ex) select job
From emp;
----> 직업이 중복되어도 그대로 나옴

Select distinct job
From emp;
----> 중복이 제거됨

006 데이터를 정렬해서 출력하기(ORDER BY)

Order by 절은 데이터를 정렬하는 절이고 **select 문장에 맨 마지막에 기술합니다.**

Ex) select ename, sal
From emp
Order by sal asc;

Order by 다음에 정렬하고 싶은 column명(정렬할 컬럼명) 쓰기, 그 다음에 어떤 방식으로 정렬할 건지(정렬방법)
Asc : 낮은 값부터 높은 값 순으로
Desc : 높은 값부터 낮은 값 순으로

* order by 절에 컬럼을 여러개를 작성할 수 있다.

```
Select ename, deptno, sal
From emp
Order by deptno asc, sal desc;
```

---> column명이 너무 길어서 귀찮다:

```
Select ename, deptno, sal
From emp
Order by 2 asc, 3 desc;
```

이렇게 해도 됨 : ename - column 1, deptno - column2, sal - column 3 가 되는 거임

007 WHERE절 배우기 1(숫자 데이터 검색)

"where절을 사용하면 특정 조건에 대한 데이터만 선별해서 출력할 수 있다"

Ex) select ename, sal
From emp
Where sal = 3000;

(검색조건/ 해당 행만 가져옴)

008 WHERE절 배우기 2(문자와 날짜 검색)

Where 절로 데이터를 검색할 때 숫자와는 다르게 문자는 양쪽에 **''(single quotation mark)를 둘러줘야 합니다.**

Ex) select ename, sal
From emp
Where ename='SCOTT';

❖ **설명 : SQL은 대소문자를 구분하지 않으나 데이터는 대소문자를 구분합니다.**
-----> scott은 무조건 대문자로 써주어야 함. 데이터가 SCOTT으로 입력되어 있으니
❖ **설명 : 싱글 쿼테이션 마크 안에 있는 데이터는 문자 또는 날짜이다 라고 오라클에게 알려주는 것임**

009 산술 연산자 배우기(*, /, +, -)

Ex) select name, sal, sal + 3000
From emp;

010 비교 연산자 배우기 1(>, <, >=,<=, =, !=,<>, ^=)

같거나 같지 않거나
!=,<>, ^= : 같지 않다

Ex) select name, sal, sal + 3000
From emp;

011 비교 연산자 배우기 2(BETWEEN AND)

~와 ~사이 나타낼 때

Ex) select name, sal
From emp
Where sal between 1000 and 3500;

같거나 같지 않거나

!=,<>, ^= : 같지 않다

012 비교 연산자 배우기 3(LIKE)

예제 : 이름의 첫글자가 S로 시작하는 사원들의 이름을 출력하시오!

Select ename
From emp
Where ename like 'S%'

- 설명 :
  like는 ~일 것 같은 이라는 영어 뜻처럼 이름의 첫번째 철자가 S로 시작할 것 같은과 같은 뜻이다.
  %는 와일드 카드라는 뜻으로 이 자리에 뭐가 와도 상관 없다.
  그 첫자의 갯수가 몇개가 돼도 관계없다는 뜻이다.

  Like와 같이 쓸 수 있는 키워드가 2개가 있다.
  %: 와일드 카드 - 이 자리에 뭐가 와도 관계없고 그 갯수가 몇 개가 되도 관계 없다.

  _ : 언더바 - 이 자리에 뭐가 와도 관계 없는데 자릿수는 한개여야 된다.

10/22 (013-018)

2020년 10월 22일 목요일 오전 9:31

<복습>

1. SQL을 왜 배워야 하는가?
   데이터를 검색하는데 파이썬의 판다스를 이용해도 되는데 왜 굳이 SQL을 해야하는가?

- 회사의 중요한 데이터는 다 database에 저장되어 있고, 그리고 대용량 데이터라서 대용량 데이터에서 데이터를 검색하는 검색 시간이 오래걸립니다. 생각 안하고 SQL을 작성하면 검색 시간이 너무 오래 걸려서 빨리 결과를 볼 수 없다. 그래서 항상 검색시간을 고려해서 SQL을 작성해야 합니다. 그런데 오라클은 버전이 업그레이드 될수록 점점 인공지능화 되어가고 있어서 SQL을 자체적으로 튜닝을 합니다.
- SQL 튜닝 : 검색시간이 빨라지게 하는 것

2. SQL을 실습 위주로 배우는 이유는?
- 일머리를 배우기 위해서
- 공부머리(여러 자격증 취득)와 일머리가 있는데 일머리는 본인이 직접 SQL을 작성해서 데이터를 검색해 낼 수 있어야한다.

3. 어제 배운 것 복습
- 기본 SELECT 문 : select 컬럼명
  from 테이블명
  where 검색조건
  order by 정렬할 컬럼명

  코딩순서 : select --> from --> where --> order by
  실행순서 : from --> where --> select --> order by

- 연결 연산자 : || --> 문자열로 결과를 출력하는 일이 종종 있기 때문에
  ex) 김인수님의 통장 총 잔고는 39,000원 입니다.

- 컬럼별칭: 결과로 출력될 컬럼의 이름을 변경하고 싶을 때 사용
  ex) ename --> 이름

  비교연산자 : > , < , >=, <=, =, !=. <>, ^=
  기타비교연산자 :
  a) between .. and
  b) like
  c) is null
  d) in

*우리반 데이터를 생성

통신사: sk, lg, kt

```
Inset into emp12
values ('김미승', 27, '여', '금융학과', 'lg', 'miseung.hailey@gmail.com',
'경기도 용인시 수지구 성복동');
```

013 비교 연산자 배우기 4(IS NULL)

NULL 값을 조회할 때 사용하는 연산자가 is null 입니다.
null 값은 데이터가 없는 상태 또는 알 수 없는 값(unknown) 입니다.

알 수 없는 값 : 오라클

Ex) 이름과 월급과 커미션을 출력하시오!
커미션 없는 사람들은 null로 표시가 됨

014 비교 연산자 배우기 5(IN)

where 절의 검색조건에서 여러개의 행을 비교할 때는 in을 사용해야 합니다.

예제 : 사원번호가 7788, 7902인 사원들의 사원번호와 이름을 조회하시오.

자주 범하는 오류:

```
select empno, ename
from emp
where empno - (7788,7902);
```

❖ 설명 : equal 연산자는 하나의 값만 비교할 수 있습니다. 여러개의 값을 비교할 때는 in을 사용해야 합니다.

정답:

```
select empno, ename
from emp
where empno in (7788, 7902);
```

015 논리 연산자 배우기(AND, OR, NOT)

오라클 연산자의 종류 3가지
1. 산술 연산자 : */+

016 대소문자 변환 함수 배우기(UPPER, LOWER, INITCAP)

함수(fuction)?
어떤 특정 기능을 구현해 놓은 코드의 집합.

오라클 연산자의 종류 3가지

1. 산술 연산자 : */+
2. 비교 연산자 : >, <, >=, <=, =, !=, <>, ^=
   기타 연산자 :
   between ... and
   like
   is null
   in

3. 논리 연산자: and, or, not
   Ex) 직업이 SALESMAN 이고 월급이 1200 이상인 사원들의 이름과 월급과 직업을 출력하시오.

```
select ename, sal, job
from emp
where job = 'SALESMAN' and sal >= 1200;
```

True and True면 True여서 결과가 출력이 되었다.
False and True면 False여서 결과가 출력이 안되었다.
False or True는 True이므로 결과가 출력이 된다.
or 작성시 신중해야 한다.

함수(fuction)?
어떤 특정 기능을 구현해 놓은 코드의 집합.

입력값 ------> 함수 ------> 출력값

함수를 사용하는 이유?
함수를 이용하면 좀 더 복잡한 데이터 검색을 할 수 있다.

Ex) 영화 겨울왕국에는 elsa가 많이 나올까 anna가 많이 나올까?

우리반 학생들이 제일 많이 쓰는 통신사가 어디인가?

함수의 종류 2가지?

1. 단일행 함수:
   문자 : upper lower, initcap
   숫자
   날짜
   변환
   일반

2. 복수행 함수: max, min, avg, sum, count, var, stddev

upper : 대문자로 출력하는 함수
lower : 소문자로 출력하는 함수
initcap: 첫번째 철자는 대문자로 출력하고 나머지는 다 소문자로 출력하는 함수

예제: select upper(ename), lower(ename), initcap(ename)
from emp;

-> 대문자, 소문자, 첫글자만 대문자 이런식으로 해서 리스트 쭉 나옴

017 문자에서 특정 철자 추출하기(SUBSTR)

subtr : 특정 부분을 잘라내는 함수

예제 : select ename, substr(ename, 1,1)
from emp;

김미승
1 2 3
-3 -2 -1

따라서 ---> 결과는 '김' 만나옴

```
select ename, substr(ename, 1, 2)
from emp;
```
결과 : '김미'까지 나옴

```
select ename, substr(ename, 2, 2)
from emp;
```
결과 : '미승'이 나옴 (2번째 자리부터 시작)

```
select ename, substr(ename, -1, 2)
from emp;
```
결과 :-1값부터 두자가 나와야하니 '승'만 나옴

```
select ename, substr(ename, -3, 2)
from emp;
```
결과 : -3값부터 두자가 나와야 하니 '김미'가 나옴

예제 : select ename, substr(ename, 1, 1)
from emp12;
성씨만 갖고오기

예제 : 성씨가 이씨인 학생들의 이름을 출력하시오.
like쓰지 말고 in과 substr을 써서 출력하시오!

```
select ename, substr(ename, 1,1)
```

018 문자열의 길이를 출력하기(LENGTH)

```
select ename, length(ename)
from emp;
```

철자의 길이가 나옴

```
from emp12
where substr(ename, 1,1) in ('ㅇ');
```

또는

```
select ename, substr(ename, 1,1)
from emp12
where substr(ename, 1,1) = 'ㅇ';
```

10/23 (019-030)

2020년 10월 23일 금요일 오전 9:40

<복습>

1. 기본 SELECT문
2. 함수
   단일행 함수 : 문자, 숫자, 날짜, 변환, 일반
   복수행 함수 : min max

019 문자에서 특정 철자의 위치 출력하기(INSTR)

"특정 철자의 자릿수를 출력하는 함수"

예제 :

1. select instr('smith;, 'm')
   from dual;

   ---> 2

2. select instr('smith;, 't')
   from dual;
   ---> 4

3. 우리반 테이블에서 이메일을 출력하고 그 옆에 이메일에서
   @가 몇번째 자리에 있는지 출력하시오!
   select email, instr(email, '@')
   from emp12;

4. 우리반 테이블에서 이메일에서 @ 앞에까지의 철자를 잘라내시오!
   select substr(email,1,instr(email,'@')-1)
   from emp12;

020 특정 철자를 다른 철자로 변경하기(REPLACE)

"특정 철자를 다른 철자로 변경하는 함수"

예제 : select replace('smith', 'm', 'k')
from dual;
-----> skith

문법 : replace (컬럼명, 대체전 문자, 대체후 문자)

021 특정 철자를 N개 만큼 채우기(LPAD, RPAD)

항상 고정된 자릿수를 보장하기 위해서 필요한 함수

문법: lpad (컬럼명, 전체 자릿수, 채워넣을 값)
l : left
r : right
pad : 채워 넣다

Ex) select sal, lpad(sal, 10, '*')
rpad(sal, 10, '*')
from emp;

설명 : lpad(sal, 10, '*')의 뜻은 월급을 출력하는데 전체 10자리 잡고 월급을 출력하고 남은 왼쪽자리에 *를 채워넣겠다 라는 뜻이다.

022 특정 철자 잘라내기(TRIM, RTRIM, LTRIM)

공백을 잘라낼 때 많이 사용하는 함수
공백 때문에 데이터 검색이 안되는 경우가 종종 있기 때문에 trim 함수를 자주 사용합니다.

ltrim : 왼쪽에 있는 공백을 잘라버리겠다.
rtrim : 오른쪽에 있는 공백을 잘라버리겠다.
trim : 양쪽에 있는 공백을 잘라버리겠다.

023 반올림해서 출력하기(ROUND)

숫자 함수

1. round : 반올림하는 함수
2. trunc : 잘라내서 버리는 함수
3. mod : 나눈 나머지 값을 구하는 함수

예제 : select round (768.567, 2)
from dual;
답 : >>> 768.57

---> 7 6 8 . 5 6 7
자릿수 -3, -2, -1, 0, 1, 2, 3

024 숫자를 버리고 출력하기(TRUNC)

select trunc(785.657, 2)
from dual;

답 : >>> 786.65

-----> 7 8 5 . 6 5 7
자릿수 0 , 1, 2, 3

설명 : trunc는 버리는 함수인데 소수점 이전은 지정된 자리를 포함해서 버리고 소수점 이후는 지정된 자리 이후부터 버린다.

설명 : 소수점 이전은 바로 그 자리에서 반올림한다.
소수점 이후 두번째 자리를 기준으로 두고 뒤에서 반올림한다.

025 나눈 나머지 값 출력하기(MOD)

Ex) select mod (10,3)
from dual;

날짜 함수

1. months_between
2. add_months
3. next_day
4. last_day

026 날짜 간 개월 수 출력하기(MONTHS_BETWEN)

날짜 - 숫자 = 날짜
날짜 + 숫자 = 날짜
날짜 - 날짜 = 숫자

** 오늘 날짜를 확인하는 방법
select sysdate
from dual;

select sysdate
from dual;
>>> 오늘날짜
select sysdate - 1
from dual;
>>>어제 날짜
select sysdate + 1
from dual;
>>> 내일 날짜
months_between(최신날짜, 옛날날짜)
날짜와 날짜 사이의 개월수를 출력

027 개월 수 더한 날짜 출력하기(ADD_MONTHS)

Ex) 오늘날짜에서 100달 뒤에 돌아오는 날짜가 몇일인가?

select add_months(sysdate, 100)
from dual;

028 특정 날짜 뒤에 오는 요일 날짜 출력하기(NEXT_DAY)

Ex) 오늘부터 앞으로 바로 돌아올 월요일의 날짜를 출력하시오!
select next_day(sysdate, '월요일')
from dual;

029 특정 날짜가 있는 달의 마지막 날짜 출력하기(LAST_DAY)

select sysdate, last_day(sysdate)
from dual;

함수 복습

1. **단일행 함수** : 문자함수, 숫자함수, 날짜함수

문자함수	숫자함수	날짜함수		
upper	round	months_between		
inticap	trunc	add_months		
instr	mod	last_day		
replace				
lpad/ rpad				
trim/ rtrim/ ltrim				

2. **복수행 함수** : max, min, avg, count, sum
(그룹함수)

***현재 내가 접속한 세션의 날짜 형식을 확인**

select *
from nls_session_parameters;

NLS_DATE_FORMAT RR/MM/DD
년도/월/일

*미국의 오라클 환경으로 날짜 형식을 변경해 본다.

alter session set nls_date_format='DD/MM/RR';

***날짜 형식**

년도 : RRRR, YYYY, RR, YY

RR	vs	YY
81 -> 1981 or 2081	년도	81 1981 or 2081
2020	현재	2020
현재년도에서 가장 가까운 년도를 선택		현재세기로 인식

월 : MM, MON
일 : DD
시간 : HH, **HH24 >> 14시, 15시 표시시 사용**
분 : MI
초 : SS
요일 : day, dy, d

```
select ename, hiredate, to_char(hiredate, 'day'), ---->화요일
                        to_char(hiredate, 'dy'), ----> 화
                        to_char(hiredate, 'd') ----> 3 (일, 월, 화/ 3번째라서)
from emp;
```

030 문자형으로 데이터 유형 변환하기(TO_CHAR)

숫자형 ---> 문자형으로 변환
날짜형 ---> 문자형으로 변환할 때 사용하는 함수

Ex) 오늘이 무슨 요일인지 출력하고 싶다면?

```
select to_char( sysdate, 'day')
  from dual;
```

- to_char를 이용해서 숫자를 문자로 형 변환하기

```
select sal, to_char(sal)
 from emp;
```

출력되는 모습(값)은 똑같은데, 숫자는 오른쪽에 붙었고, 문자는 왼쪽에 붙음

```
select sal, to_char(sal, '999,999')
 from emp;
```

❖ 설명 : to_char(sal, '999,999')의 의미는 sal 숫자를 문자로 출력하는데 to_char(숫자형 컬럼, '문자포멧')의 문법에 따라 문자 포멧에 맞는 문자로 출력하는 것입니다.
9는 자릿수를 의미하고 이 자리에 0-9사이의 숫자 중에 어떤 숫자가 와도 관계는 없지만 자릿수는 한자리여야 한다는 의미입니다.

10/24 (031-036)

2020년 10월 26일 월요일 오전 9:30

복습:

1. SQL을 왜 배워야 하는지
2. 기본 SELECT문
3. 함수

- 단일행 함수 : 문자, 숫자, 날짜, 변환, 일반
- 복수행 함수 : max, min, avg, sum, count

문자함수 : upper, lower, initcap, substr, instr
length, lpad, rpad, trim, ltrim, rtrim,
replace,

숫자함수 : round, trunc, mod
날짜함수 : month_between, add_months,
next_day, last_day
변환함수 : to_char: 문자로 형변환하는 함수
to_number: 숫자로 형변환하는 함수
to_date: 날짜로 형변환하는 함수

031 날짜형으로 데이터 유형 변환하기(TO_DATE)

" 날짜로 형 변환하는 함수 "

Ex) 영국식으로 날짜 변환하고 싶을 때, 형식 맞춰주기:

```
select ename, hiredate
    from emp
    where hiredate = to_date('81/11/17, 'RR/MM/DD')';
          Δ
          날짜형 데이터 유형 = 날짜 (날짜로 맞춰줘야 함)
          이렇게 명시 해주면 어떠한 상황 속에서도 검색이 됨.
          for example, where hiredate = '81/11/17'이라고 쓸 때,
          시스템 상의 날짜가 RR/MM/DD가 아니면 오류가 남.

    where sal = 3000;
          Δ
          숫자형 데이터 유형 = 숫자 (숫자로 맞춰줘야 함)
```

❖ 설명 : 날짜형 데이터를 검색할 때는 반드시
to_date 함수를 사용할 것을 권장합니다.

032 암시적 형 변환 이해하기

**** to_number**
" 숫자로 형 변환하는 함수"

월급이 3000인 사원의 이름과 월급을 출력하시오

```
select ename, sal
  from emp
  where sal  =      3000;
        Δ숫자형    Δ숫자형

select ename, sal
  from emp
  where sal  =      '3000';
        Δ숫자형    Δ문자형
                  (내부적으로 문자형을 숫자형으로 변환합니다.)
```

이렇게 해도 실행이 될까?
>>>>된다.

다른 DATABASE 소프트웨어에서는 에러가 나면서 실행이 안되지만
오라클에서는 가능하다.
오라클이 알아서 문자형을 숫자형으로 변환합니다.
but 에러는 안나지만 검색속도가 현저히 느려진다.

```
>>>>
set autot on

select ename, sal
  from emp
  where sal = '3000';
```

이렇게 확인해보면:

```
1 - filter("SAL"=3000) 이렇게 나옴.
1 - filter(TO_NUMBER("SAL")=2000)
```

>>> oracle이 내부적으로 수행한 것

033 NULL 값 대신 다른 데이터 출력하기(NVL, NVL2)

일반함수 :

1. nvl 함수
2. decode 함수
3. case 함수

NVL 함수 : null 값 대신에 다른값을 출력하고 싶을 때 사용하는 함수

예제 : 이름, 월급, 커미션, 월급 + 커미션을 출력하시오.

```
select ename, sal, comm, sal + comm
    from emp;
```

>>> comm에 null 값 존재
null 값 : 데이터가 없는 상태 또는 알 수 없는 값

```
select ename, sal, comm, nvl(comm, 0), sal+nvl(comm, 0)
    from emp;
```

이렇게 하면 문제 해결

❖ 설명 : nvl 함수로 다른값으로 대체해서 출력하는 것이지 실제로 테이블의 데이터가 0으로 변경된 것은 아니다.

034 IF문을 SQL로 구현하기 1(DECODE)
035 IF문을 SQL로 구현하기 2(CASE)

"If 문을 SQL로 구현할 때 사용하는 함수"

decode: 코딩 안하고 함수로 구현하겠다.. 라는 뜻

만약에 무슨일이 벌어지면 어떻게 행동해라 라고 컴퓨터 프로그래밍을 하는 것

```
Ex ) select ename, sal, deptno, decode( deptno, 10, 5600
                                                20, 4500,
                                              0)  as 보너스
     from emp;
```

1 - filter(TO_NUMBER("SAL")=2000)

>>> oracle이 내부적으로 수행한 것

- 설명 : where절에 검색 조건을 적을 때 주의할 사항은 문자 컬럼의 데이터를 검색할 때는 문자로 검색하고 숫자컬럼의 데이터를 검색할 때는 숫자로 검색해야 한다.
- 만약에 문자형인데 숫자형으로 검색하거나 숫자형인데 문자형으로 검색할 시 오라클은 에러는 나지 않지만 검색성능이 느려지게 된다.
- >>>반드시 검색조건을 적을 때 위의 사항을 지켜줘야 합니다.

이 뜻이 문자형 >>> 숫자형으로 바꼈다는 것을 의미함.
SAL을 문자형(varchar)으로 입

desc emp; 하면 :

```
이름        널?       유형
--------- -------- ------------
EMPNO              NUMBER(4)
ENAME              VARCHAR2(10)
JOB                VARCHAR2(9)
MGR                NUMBER(4)
HIREDATE           DATE
SAL                NUMBER(7,2)
COMM               NUMBER(7,2)
DEPTNO             NUMBER(2)
```

숫자형 : NUMBER
문자형 : VARCHAR
날짜형 : DATE

Ex) select ename, sal, deptno, decode(deptno, 10, 5600
20, 4500,
0) as 보너스
from emp;

>>> 부서번호가 10번이면 5600을 출력하고
부서번호가 20번이면 4500을 출력하고
나머지 부서번호는 0을 출력해라.

결과 >>> ename sal deptno 보너스 이렇게 컬럼이 나옴

- 중요설명 : decode는 등호(=)비교만 가능하기 때문에 부등호 비교를 하려면 case문을 사용해야 한다.
case문은 등호비교, 부등호 비교 둘 다 가능합니다.

```
select ename, sal, case when sal >=4000 then 500
                        when sal >=2000 then 300   (자동으로 2000~4000 사이로 등록이 됨)
                        else 0 end
from emp;
```

>>>end로 꼭 마무리를 지어줄 것

036 최대값 출력하기(MAX)

함수의 종류
1. 단일행 함수
2. 복수행 함수 :

Ex) 최대 월급을 출력하시오.

```
 select max(sal)
  from emp;
```

>>> 5000

```
select deptno, max(sal)
    from emp
    where deptno = 30
  group by deptno;
```

- 설명 : group by deptno를 하게 되면 여러 개 나오려는 deptno를 grouping 해준다.

>>> max는 한가지 값만 내려고 하고, deptno는 여러 값이 있기 때문에 group by를 해주지 않으면 오류가 난다.

10/29 (037-042)

2020년 10월 29일 목요일 오전 9:46

- 복습

1. SQL을 왜 배워야 하는지?
2. 기본 select문

	코딩순서	실행순서	
select	1	5	검색할 컬럼명
from	2	1	검색할 테이블명
where	3	2	검색조건
group by	4	3	그룹핑할 컬럼
having	5	4	그룹함수로 만든 조건
order by	6	6	정렬할 컬럼명

6가지 절 (데이터 분석가 취직 시 단골문제)

3. 함수
   단일행 함수 : 문자, 숫자, 날짜, 변환, 일반
   복수행 함수 : max, min, avg, sum, count

037 최소값 출력하기(MIN)

"최소 값 출력하기"

038 평균값 출력하기(AVG)

" avg 함수로 평균값을 출력할 수 있습니다."

039 토탈값 출력하기(SUM)

"토탈값을 출력하는 함수"

Ex) 사원 테이블의 총 월급을 출력하시오.

```
select sum(sal)
    from emp;
```

❖ 설명 : group 함수로 조건을 줄 때는 where절에 사용하면 안되고 having절에 사용해야 합니다.
where절에는 그룹함수를 사용하지 않은 일반적인 검색조건을 줄 때 사용합니다.

040 건수 출력하기(COUNT)

"건수를 출력하는 함수"

Ex) 사원테이블의 인원수를 출력하시오!

```
select count(empno)
 from emp;
```

>>> 14 / empno 있는 부분만 카운트

```
select count(*)
    from emp;
```

>>>14 / 전체 행을 하나씩 카운트

```
select count(comm)
    from emp;
```

>>> 4 / comm값이 있는 부분만 카운트

"그룹함수는 null값을 무시한다"

041 데이터 분석 함수로 순위 출력하기 1(RANK)

"데이터 분석 함수 : 데이터 분석을 용이하게 하기 위해서 제공하는 함수"

그중에 rank는 순위를 출력하는 함수 입니다.

Ex) 이름, 월급, 월급에 대한 순위를 출력하시오.

```
select ename, sal, rank() over (order by sal desc) as 순위
    from emp;
```

over : 확장하다

042 데이터 분석 함수로 순위 출력하기 2(DENSE_RANK)

Ex)

```
select lower(substr(telecom,1,2)), ename, age, dense_rank () over
(partition by lower(substr(telecom,1,2))
                                    order by age desc) as 순위
    from emp12;
```

dense : 밀집하다

rank : 순위

10/30 (043-053)

2020년 10월 30일 금요일 오전 9:38

복습

- 함수 :

- 단일행 함수: ---------->함수 ---------->
  한개의 행 한개의 행

  문자함수 : upper, lower, initcap, substr, instr, lpad, rpad, trim, ltrim, rtrim, replace
  숫자함수 : round, trunc, mod
  날짜함수 : months_between, add_months, next_day, last_day
  변환함수 : to_char, to_number, to_date
  일반함수 : decode (등호비교만), case(등호, 부등호 모두), nvl, nvl2

- 복수행 함수: max, min, avg, sum, count
  (그룹함수)
  ---------->
  ----------> 함수 --------->
  ---------->
  :
  :
  여러개의 행 한개의 행

- 데이터 분석 함수
  회사의 전산 시스템의 구조 (데이터 저장 부분에 대한 서버의 종류)

OLTP 서버	**DW서버**
Online Transaction Processing	Data Warehouse
실시간으로 봐야할 데이터들을 저장	과거 이력 데이터
Ex) 지금 내 현재 통장 잔고 데이터	Ex) 10년전 과거 내 통장 잔고 데이터
	>>> 데이터 분석함수 사용

043 데이터 분석 함수로 등급 출력하기(NTILE)

"등급을 출력하는 함수" / 제조업에서 많이 사용

```
Ex ) select ename, sal, ntile(4) over (order by sal desc) 등급
       from emp;
```

❖ 설명 : 월급이 높은 순으로 정렬한 데이터를 4등급으로 나누겠다는 뜻
~25% : 1등급
25~50% : 2등급
50~75% : 3등급
75%~100% : 4등급

044 데이터 분석 함수로 순위의 비율 출력하기(CUME_DIST)

"순위에 대한 비율을 출력하는 데이터 분석함수"

```
Ex ) select ename, sal,
            cume_dist() over (order by sal desc) 비율
       from emp;
```

cume_dist() over (정렬하고 싶은 부분)

045 데이터 분석 함수로 데이터를 가로로 출력하기(LISTAGG)

"데이터를 가로로 출력하는 함수"

```
Ex)
      select deptno, listagg(ename, ', ') within group (order by ename asc) 이름
      from emp
      group by deptno;
```

```
>>>
10      CLARK, KING, MILLER
20      ADAMS, FORD, JONES, SCOTT, SMITH
30      ALLEN, BLAKE, JAMES, MARTIN, TURNER, WARD
```

❖ 설명 : 이름을 가로로 출력하는데 콤마(,)로 구분해서 출력하겠다.
이름이 abcd 순으로 정렬되어서 출력되고 있다.
listagg는 다른 분석함수와는 다르게 group by 절이 필요하다.

046 데이터 분석 함수로 바로 전 행과 다음 행 출력하기(LAG, LEAD)

"바로 전행을 옆에 나오게 하거나 바로 다음행을 옆에 나오게 할 때 사용하는 함수"

```
Ex ) select ename, sal, lag(sal,1) over ( order by sal asc) as 전행,
                        lead(sal,1) over (order by sal asc) as 다음행
       from emp;
```

047 COLUMN을 ROW로 출력하기 1(SUM+DECODE)

회사에서 데이터를 저장하는 서버 종류 2가지

048 COLUMN을 ROW로 출력하기 2(PIVOT)

"세로를 가로로 출력하는 함수"

047 COLUMN을 ROW로 출력하기 1(SUM+DECODE)

회사에서 데이터를 저장하는 서버 종류 2가지

OLTP 서버 DW 서버

컬럼을 로우로 출력하는 sum+decode

Ex 1) 부서번호, 부서번호별 토탈월급을 출력하시오.

```
>>>
select deptno, sum(sal)
   from emp
   group by deptno;
```

>>>세로출력 되어서 나옴

세로출력:

```
DEPTNO  SUM(SAL) <--- 컬럼명
30          9400
10          8750
20          10875
```

가로출력:

```
10      20      30      <--- 컬럼명
8750   10870   9400   <--- 데이터
```

EX 2) 부서번호, 부서번호가 10번이면 월급이 출력되게 하고 아니면 0이 출력되게 하시오.

```
select deptno, decode(deptno, 10,sal, 0)
   from emp;
>>>
DEPTNO  DECODE(DEPTNO, 10, SAL, 0) <<<< 컬럼명
10          5000
30          0
10          2450
20          0
30          0
30          0
30          0
30          0
30          0
20          0
20          0
20          0
20          0
10          1300
```

EX 3) 위의 결과에서 부서번호 컬럼은 안 나오게 하시오!

```
select decode(deptno, 10,sal, 0)
   from emp;
```

EX 4) 위에서 출력된 14개의 데이터를 다 SUM하시오.

```
select sum(decode(deptno, 10,sal, 0))
   from emp;
```

EX 5) 위의 컬럼명을 컬럼 별칭을 써서 숫자 10으로 변경하시오

```
select sum(decode(deptno, 10,sal, 0)) as "10"
   from emp;
```

```
10 <<< 컬럼명
8750
```

오라클에서 더블 쿼테이션 마크를 사용해야 하는 경우

- 컬럼별칭 사용할 때 특수문자, 공백문자, 대소문자 구분, 숫자를 사용할 때

EX 6) 그러면 아래와 같이 20번과 30번도 그 옆에 출력하시오.

```
select sum(decode(deptno, 10,sal, 0)) as "10" ,
       sum(decode(deptno, 20,sal, 0)) as "20",
       sum(decode(deptno, 30,sal, 0)) as "30"
   from emp;
```

```
>>>
10      20      30   <<< 컬럼명
8750   10875  9400 <<< 데이터
```

048 COLUMN을 ROW로 출력하기 2(PIVOT)

"세로를 가로로 출력하는 함수"

Ex) select * <--- pivot 문 사용할 때는 그냥 *를 쓴다.
from (select deptno, sal from emp)
Δ 사원 테이블에서 그냥 부서번호와 월급만 가져온다.
emp테이블명만 딱 쓸 수 없다.
결과를 보기 위해서 필요한 컬럼만 선별해서 가져와야 됩니다.
emp 테이블명만 쓰면 에러가 납니다.
아래의 결과를 보기 위해서 직업이 필요하지도 않고 입사일이 필요하지도 않습니다.
그냥 부서번호하고 월급만 있으면 됩니다.
그리고 from 절에는 그룹함수를 쓰면 안됩니다. 에러납니다.
from 절에는 raw data만 써 주어야 합니다.
pivot (sum(sal) for deptno in (10, 20, 30)) ;

sum(sal) : 토탈월급을 출력하겠다.
for : 어떤 토탈월급?
detpno : 부서번호를 위한
in : 어떤 부서번호?
pivot : 가로로

```
>>> 10      20      30     <--- 컬럼명
    8750   10870   9400   <--- 데이터
```

049 ROW를 COLUMN으로 출력하기(UNPIVOT)

pivot : 세로 ----> 가로
unpivot : 가로 ----> 세로

*order2 라는 테이블을 생성하고 데이터를 3건 입력한다.

1. 데이터 생성 스크립트

```
create table order2
( ename   varchar2(10),
  bicycle   number(10),
  camera    number(10),
  notebook   number(10) );
```

2. 데이터 입력 스크립트

```
insert  into  order2  values('SMITH', 2,3,1);
insert  into  order2  values('ALLEN',1,2,3 );
insert  into  order2  values('KING',3,2,2 );
```

commit;

3. order2 테이블을 검색해본다

```
select *
from order2;
```

```
ENAME  BICYCLE CAMERA  NOTEBOOK
---------------------------------------
SMITH     2         3         1
ALLEN     1         2         3
KING      3         2         2
```

Ex) select *
from order2
unpivot(건수 for 물품 in (BYCICLE, CAMERA, NOTEBOOK));

```
ENAME   물품          건수
-----------------------------------
SMITH BICYCLE          2
SMITH CAMERA           3
SMITH NOTEBOOK         1
ALLEN BICYCLE          1
ALLEN CAMERA           2
ALLEN NOTEBOOK         3
KING   BICYCLE         3
KING   CAMERA          2
KING   NOTEBOOK        2
```

❖ 설명 : 가로(컬럼) ---> 세로 (데이터)
위의 SQL에서 건수와 물품은 SQL작성자가 마음대로 이름을 명명해도 되는데 이름 짓는 대로 그대로 컬럼명이 출력이 된다. 그리고 BICYCLE, CAMERA, NOTEBOOK은 양쪽에 싱글 쿼테이션 마크를 둘러주지 않아도 됩니다.

8750 10873 9400 <<< 데이터

글 쿼테이션 마크를 둘러주지 않아도 됩니다.

050 데이터 분석 함수로 누적 데이터 출력하기(SUM OVER)

"데이터를 누적해서 합계하는 데이터 분석 함수"

Ex) 사원번호, 이름, 월급, 월급의 누적치를 출력하시오.

```
select empno, ename, sal, sum(sal) over (order by empno asc) 누적치
    from emp;
```

051 데이터 분석 함수로 비율 출력하기(RATIO_TO_REPORT)

"자기의 월급이 전체 월급중에서의 비율이 어떻게 되는지 확인하는 함수"

```
Ex ) select ename, sal, ratio_to_report(sal) over() as 비율
        from emp
        where job = 'SALESMAN'
```

052 데이터 분석 함수로 집계 결과 출력하기 1(ROLLUP)

"집계한 결과를 맨 아래쪽에 출력하고 싶을 때 사용하는 함수"

Ex) 부서번호, 부서번호별 토탈월급을 출력하시오! (세로 출력)

```
select deptno, sum(sal)
    from emp
    group by deptno;
---------
select deptno, sum(sal)
    from emp
    group by rollup(deptno);
```

>>>맨 아래에 합계가 나온다.

Ex) 직업, 직업별 토탈월급을 출력하시오 (세로 출력)

```
select job, sum(sal)
    from emp
    group by job;
```

Ex) 위의 결과에서 직업별 토탈월급의 합계를 맨 아래에 출력하시오!

```
select job, sum(sal)
    from emp
    group by rollup(job);
```

053 데이터 분석 함수로 집계 결과 출력하기 2(CUBE)

"집계 결과를 위쪽에 출력하는 함수"

```
Ex ) select job, sum(sal)
        from emp
        group by cube (job);
```

11/2 (054-058)

2020년 11월 2일 월요일 오전 9:22

<복습>

1. **기본 select문장 6가지절 (실행순서)**
   5 select 보고싶은 컬럼명
   1 from 데이터가 들어있는 테이블명
   2 where 검색조건
   3 group by 그룹핑할 컬럼명
   4 having` 그룹함수로 만든 검색 조건
   6 order by 정렬할 컬럼명

2. **함수: 단일행함수 : 문자, 숫자, 날짜, 변환, 일반**
   복수행함수 : max, min, avg, sum, count
   (그룹함수)

   그룹함수의 특징 ?
   - null 값을 무시한다.
   - where절의 조건이 거짓이어도 결과를 리턴한다.
     Ex) select sum(sal)
     from emp
     where 1=2; <<< where절 조건이 거짓

     null값이 나옴 >>>null값이라도 결과를 출력해 줌

     select sal
     from emp
     where 1=2;
     이렇게 조건을 주면 값이 안나온다
     (그룹함수가 아니어서)

     그룹함수 작성시에는 where절의 조건을 잘 명시해야 합니다.

3. **데이터 분석 함수 :**
   1. **rank : 순위 출력**
   2. **dense_rank : 순위 출력**
   3. **ntile : 등급 출력**
   4. **cume_dist : 비율 출력**
   5. **listagg : 데이터를 가로로 출력**
   6. **sum(컬럼명) over (문법) : 누적 데이터 출력**
   7. **ratio_to_report : 비율 출력**
   8. **rollup : 집계 결과를 맨 아래에 출력**
   9. **cube : 집계 결과를 맨 위에 출력**

데이터 분석함수를 이용하면 우리가 현업에서 필수로 검색해야 하는 데이터를 긴 SQL로 작성하지 않고 간단한 함수를 이용해서 볼 수가 있습니다.

- 오라클 버전의 히스토리:
  8 -> 8i -> 9i -> 10g -> 11g -> 12c -> 18c -> 19c
  i : internet
  g : grid - 성능이 보통인 여러개의 컴퓨터를 여러대 붙여서 마치 성능이 좋은 큰 서버처럼 운영하는 기술
  c: cloud

054 데이터 분석 함수로 집계 결과 출력하기 3(GROUPING SETS)

"집계결과를 출력하는 데이터 분석 함수입니다"

Ex) select deptno, sum(sal)
from emp
group by grouping sets(deptno, ()) ;
Δ Δ
부서번호 전체 (괄호 열고 닫고)
select문에 썼던 기준중에서 써야함

select deptno, sum(sal)
from emp
group by grouping sets(deptno) ;
>>>전체 sum값이 안 나옴

055 데이터 분석 함수로 출력 결과 넘버링 하기(ROW_NUMBER)

"출력결과의 행 앞에 숫자를 차례대로 부여하는 함수입니다"

Ex) select row_number() over (order by empno), empno, ename from emp;

❖ 이렇게 row_number() 함수를 사용하면 언제 유용하냐면 출력되는 행의 맨위의 3개의
행만 가져와라 또는 맨 위의 하나의 행만 가져와라 라고 검색할 때 유용하다.
인라인뷰(from 절의 서브쿼리)를 배우면 검색 할 수 있습니다.

056 출력되는 행 제한하기 1(ROWNUM)

"rownum을 이용하면 출력되는 행의 갯수를 제한할 수 있습니다"

Ex) select rownum, empno, ename, sal
from emp;

❖ 설명 : row_number() 함수와는 다르게 인라인뷰(from 절의 서브쿼리)를 사용하지 않고도 위의 3개의 행만 가져와라 같은 데이터 검색을 할 수 있다.

select rownum, empno, ename, sal

057 출력되는 행 제한하기 2(Simple TOP-n Queries)

"order by 절까지 사용해서 검색하는 SELECT문의 출력되는 행의 일부를 가져올 때 사용하는 문법"

Ex) select rownum, empno, ename, sal
from emp
order by sal desc;

❖ 설명 : order by가 수행되기 전에 rownum이 부여되어서
번호가 뒤죽박죽 섞였다.

할 수 있다.

```
select rownum, empno, ename, sal
       from emp
       where rownum <= 3;
```

대용량 테이블의 데이터가 있는 테이블의 내용을 살짝 보고 싶다면 rownum을 사용해서 몇건의 데이터만 살펴 보시오!

❖ 설명 : order by가 수행되기 전에 rownum이 부여되어서 번호가 뒤죽박죽 섞였다.

Ex) 사원 테이블에서 월급이 높은 사원 4명만 출력하시오. 이름과 월급을 출력하시오.

```
select rownum, ename, sal
    from emp
    where rownum <=4
    order by sal desc;
```

❖ 설명 : 제대로 출력되지 않았다. 왜냐하면 실행순서 때문입니다.
emp 테이블에서 맨위의 4명만 가져와서 그 4명만으로 월급이 높은 순서대로 정렬했기 때문입니다. 위의 결과를 제대로 보려면 아래와 같이 top n query를 작성해야 합니다.

(실행순서)

```
4       select ename, sal
1          from emp
2          order by sal desc
3          fetch first 4 rows only;
```

❖ 설명 : fetch는 검색으로 가져오는 데이터를 가져와라 라는 뜻이다.
first 4 rows only는 문법인데 그 중에 4개의 행만 가져와라~ 라는 뜻입니다.

058 여러 테이블의 데이터를 조인해서 출력하기 1(EQUI JOIN)

" 조인을 이용하면 두개 이상의 테이블들의 컬럼들을 하나의 결과로 모아서 볼 수가 있습니다"

Ex) select *
from dept;

```
DEPTNO  DNAME          LOC
--------------------------------------------------
10      ACCOUNTING     NEW YORK
20      RESEARCH       DALLAS
30      SALES          CHICAGO
40      OPERATIONS     BOSTON
```

DEPTNO : 부서번호
DNAME : 부서명
LOC : 부서위치

Ex) 이름, 부서위치를 출력하시오!

```
select ename, loc
       from emp, dept;
```

>>> 14 x 4건 = 56건이 되어서 전 직원이 전 위치에서 근무하는걸로 즉, 다 조인한 결과가 나옵니다.

```
select ename, loc
    from emp, dept
    where emp.deptno = dept.deptno;   <--- 조인 조건을 where 절에 기술해야 정확한 결과를 볼 수 있습니다.
```

이렇게 해주면 원하는 결과를 도출할 수 있습니다.

❖ 설명 : 조인조건을 줘야 조인을 할 수 있습니다.
"emp테이블의 deptno는 dept 테이블에 deptno와 같습니다"라고 오라클에게 알려주면서 이 컬럼으로 두개의 테이블을 서로 조인하는 것입니다.

11/3 (059-066)

2020년 11월 3일 화요일 오전 9:33

<복습>

1. 기본 select문장 6가지 절 :
   Select
   From
   Where
   Group by
   Having
   Order by
2. 함수 :
   a. 단일행함수: 문자, 숫자, 날짜, 변환, 일반
   b. 복수행 함수: max, min, avg, sum, count

   현업에서 많이 보는 검색결과들은 주로 데이터 분석함수를 이용한 결과들이 많았다.

3. 데이터 분석 함수 :
   1) rank
   2) dense_rank
   3) ntile
   4) cume_dist
   5) listagg
   6) report_to_ratio
   7) lag, lead
   8) sum(컬럼명) over (문법)
   9) rollup
   10) cube
   11) grouping sets

4. 조인(join) 문장

   " 하나의 테이블에서 얻을 수 있는 정보가 아닌 여러개의 테이블에서 얻을 수 있는 정보를 하나의 결과로 보여주기 위해서 만든 문법"

   Ex) select e.ename, d.loc <<< 컬럼명
   from emp e, dept d <<< 테이블명
   where e.deptno = d.deptno and d.loc = 'DALLAS';
   조인조건 검색조건
   emp<---------->dept (조인조건이 연결고리가 됨)

059 여러 테이블의 데이터를 조인해서 출력하기 2(NON EQUI JOIN)

"조인하려는 두개의 테이블 사이에 공통된 컬럼이 없었을 때 사용하는 조인 문법"

SALGRADE 테이블 만들기

```
LOSAL GRADE HISAL
----------------------
700    1      1200
1201   2      1400
1401   3      2000
2001   4      3000
3001   5      9999
```

Ex) 이름, 월급, grade(급여등급)을 출력하시오!

```
emp ----------salgrade

select ename, sal, grade
    from emp, salgrade
    where 조인조건
>>>>
select e.ename, e.sal, s.grade
  from emp e, salgrade s
  where e.sal between s.losal and s.hisal;
```

❖ 설명 : "조인 문법의 종류 2가지"

1. 오라클 조인 문법
   a. equi join : 두개의 테이블 사이에 공통된 컬럼이 있었을 때
   b. non equi join : 두개의 테이블 사이에 공통된 컬럼이 없을 때

060 여러 테이블의 데이터를 조인해서 출력하기 3(OUTER JOIN)

(속도가 느려짐 - 쓰는것 자제하기)

"조인하려는 두 테이블의 공통된 컬럼인 deptno의 데이터가 서로 똑같이 일치하지 않을 때 조인하기 위해서 사용하는 조인 컬럼"

Ex)사원 테이블에서 부서번호를 출력하는데 중복제거해서 출력하시오!

```
select distinct deptno
    from emp;
```

Ex) 부서(dept)테이블에서 부서번호를 출력하시오.

```
select deptno
    from dept;
```

>>>안나옴

```
select e.ename, d.loc
    from emp e, dept d
    where e.deptno(+) = d.deptno;
                  아우터조인 싸인(모자란쪽에 붙여줌)
```

❖ 설명 : outer join sign (+)는 결과로 출력될 때 데이터가 모자란 쪽에 붙여준다. emp테이블에서는 부서번호가 10, 20, 30이 있고 40번이 없습니다.
dept 테이블에서는 부서번호가 10, 20, 30, 40이 있습니다.
하지만 양쪽에 다 (+) 아우터 조인 싸인을 붙이면 오류 납니다.

c. outer join : 두개의 테이블에 공통된 컬럼은 있으나 조인하려는 컬럼에 데이터가 서로 일치하지 않을 때

d. self join : 자기 자신의 테이블과 조인하는 조인

2. 1999 ansi 조인문법
   i. ansi : american national standard institute

061 여러 테이블의 데이터를 조인해서 출력하기 4(SELF JOIN)

"자기 자신의 테이블과 조인하는 조인 문법"

- 왜 자기 자신의 테이블과 조인을 해야하는가?
  사원 테이블로 예를 들면 사원이름과 그 사원을 관리하는 관리자의 이름을 하나의 결과로 볼 수 있기 때문입니다.

Ex) 사원번호, 사원이름, 관리자 번호(mgr)을 출력하시오!

```
select empno, ename, mgr
    from emp;
```

```
EMPNO ENAME      MGR
--------------------------------
7839      KING
7698      BLAKE      7839
7782      CLARK      7839
7566      JONES      7839
7654      MARTIN     7698
7499      ALLEN      7698
7844      TURNER     7698
7900       JAMES     7698
7521       WARD      7698
7902       FORD      7566
7369       SMITH     7902
7788        SCOTT    7566
7876        ADAMS    7788
7934        MILLER   7782
```

SMITH (7369) --> FORD (7902) --> JONES(7566) ---> KING(7839)

Ex) 사원이름, 해당 사원의 관리자 이름을 출력하시오!
(이름 두번 출력)

```
select 사원.ename, 관리자.ename
   from emp 사원, emp 관리자
   where 사원.mgr =관리자.empno;
```

062 여러 테이블의 데이터를 조인해서 출력하기 5(ON절)

- **조인문법 2가지?**
  1. **오라클 조인 문법**
     - **equi join**
     - **non equi join**
     - **outer join**
     - **self join**
     4가지 조인문법을 모두 잘 알고 있어야 함
  2. **1999 ANSI 조인 문법**
     - **on절을 사용한 조인**
     - **using절을 사용한 조인**
     - **natural join**
     - **left/right/full outer 조인**
     - **cross조인**
     On절을 사용한 조인 문법과 아우터 조인문법을 잘 알고 있어야 합니다.

Ex)

1. 오라클 equi join

```
select e.ename, d.loc
      from emp e, dept d
      where e.deptno = d.deptno;
```

2. on절을 사용한 조인

```
select e.ename, d.loc
      from emp e join dept d
      on (e.deptno = d.deptno);
---------------------------------------
select e.ename, e.sal, d.loc
   from emp e join dept d
   on (e.deptno = d.deptno) <<< 조인조건
   where e.sal >= 2400; <<< 검색조건
```

❖ 설명 : on절을 사용한 조인 문법은 조인조건은 on절에 주게 되어있고 검색조건은 where절에 주게끔 구분해 놓았다. 이렇게 해야 에러가 안난다.

063 여러 테이블의 데이터를 조인해서 출력하기 5(USING절)

```
Ex ) select e.ename, d.loc
      from emp e join dept d
      using  (deptno);
```

>>> 연결고리가 되는 컬럼명 테이블 별칭 없이 써준다.

064 여러 테이블의 데이터를 조인해서 출력하기 6(NATURAL JOIN)

```
Ex ) select e.ename, d.loc
      from emp e natural join dept d;
```

❖ 설명 : where절 없이 간단하게 조인하는 조인 문법
오라클이 알아서 두 테이블 사이에 공통된 컬럼이 있는지 찾아보고 조인한다.

065 여러 테이블의 데이터를 조인해서 출력하기 7(LEFT/RIGHT OUTER JOIN)

Ex)

1. 오라클 조인 문법 (아우터 조인)

```
select e.ename, d.loc
      from emp e, dept d
```

066 여러 테이블의 데이터를 조인해서 출력하기 8(FULL OUTER JOIN)

```
Ex ) select e.ename, d.loc
      from emp e, dept d
      where e.deptno(+) = d.deptno (+);
```

1. 오라클 조인 문법 (아우터 조인)

```
select e.ename, d.loc
      from emp e, dept d
      where e.deptno(+) = d.deptno;
```

2. 1999 ANSI 조인 문법

```
select e.ename, d.loc
      from emp e right outer join dept d
      on (e.deptno = d.deptno);
```

(삽입하기)

```
insert into emp(empno, ename, sal, job, deptno)
   values ( 1221, 'jack', 3500, 'SALESMAN', 70 );

commit;
```

오라클 조인문법:

```
select e.ename, d.loc
      from emp e, dept d
      where e.deptno = d.deptno(+)
```

❖ 설명 : dept테이블에는 70번 부서가 없으므로 조인할 때 equi조인을 하게 되면 결과가 안나오고 outer조인을 사용해야 합니다.

- **1999 ANSI 문법의 cross 조인이란?**

where절 없이 조인해서 전체를 다 조인하는 조인 문법입니다.

1. 오라클 조인 문법

```
select e.ename, d.loc
      from emp e, dept d;
```

2. 1999 ANSI문법

```
select e.ename, d.loc
      from emp e cross join dept d;
```

```
from emp e, dept d
where e.deptno(+) = d.deptno (+);
```

>>>오류난다.

위의 결과 출력을 가능하게 해주는 조인 문법이 1999 ANSI의 full outer join입니다.

```
select e.ename, d.loc
      from emp e full outer join dept d
      on (e.deptno = d.deptno);
```

결과:

```
ENAME    LOC
-------------------
KING       NEW YORK
BLAKE      CHICAGO
CLARK      NEW YORK
JONES       DALLAS
MARTIN     CHICAGO
ALLEN      CHICAGO
TURNER     CHICAGO
JAMES      CHICAGO
WARD       CHICAGO
FORD        DALLAS
SMITH       DALLAS
SCOTT       DALLAS
ADAMS      DALLAS
MILLER     NEW YORK
jack          null
null          BOSTON
```

이렇게 다 나온다

11/4 (067-074)

2020년 11월 4일 수요일 오전 9:29

067 집합 연산자로 데이터를 위아래로 연결하기 1(UNION ALL)

- 데이터를 연결해서 출력하는 방법 가지

1. 조인(join) : 데이터를 양 옆으로 연결해서 출력하는 방법
2. 집합연산자 : 데이터를 위아래로 연결해서 출력하는 방법

- 집합연산자의 종류 4가지

1. union all
2. union
3. intersect
4. minus

Ex) 직업, 직업별 토탈월급을 출력하시오. (세로출력)

```
select job, sum(sal)
    from emp
    group by job;
```

```
JOB              SUM(SAL)
---------------------------
SALESMAN     5600
CLERK          4150
ANALYST       6000
MANAGER     8275
PRESIDENT     5000
```

Ex) 전체 토탈월급을 출력하시오!

```
>>>>>
select sum(sal)
    from emp;
```

```
SUM(SAL)
-----------
29025
```

```
>>>>
select '전체토탈:' as job,sum(sal)
    from emp;
```

```
JOB              SUM(SAL)
---------------------------
전체토탈:       29025
```

Ex) 위의 SQL을 하나로 합쳐서 데이터가 위아래로 출력되게 하시오!

```
select job, sum(sal)
    from emp
    group by job
union all <<< 데이터 접착제 역할
select '전체토탈:' as job, sum(sal)
    from emp;
```

※ 설명 : union all로 위의 쿼리문의 결과와 아래의 쿼리문의 결과를
하나로 합쳐서 출력하고 있다.
※ 집합 연산자를 사용할 때 주의사항은 다음과 같다.※

1. 집합 연산자 위 아래의 쿼리문의 컬럼의 갯수가 동일해야 합니다.
2. 집합 연산자 위 아래의 쿼리문의 컬럼의 데이터 타입(문자형 or 숫자형)도 동일해야 합니다.
3. 집합 연산자 위 아래의 쿼리문의 컬럼의 컬럼명이 동일해야 합니다.
4. order by 절은 맨 아래 쿼리문에만 작성할 수 있습니다.

068 집합 연산자로 데이터를 위아래로 연결하기 2(UNION)

"union은 union all과 같은 합집합 연산자인데 차이점은
union은 order by절을 사용하지 않아도 정렬을 암시적으로 수행합니다.
그리고 중복된 데이터를 하나로 출력합니다."

Ex)

```
select to_char(hiredate, 'RRRR') as hire_year, sum(sal)
    from emp
    group by to_char(hiredate, 'RRRR')
union
select '토탈:' as hire_year, sum(sal)
    from emp;
```

069 집합 연산자로 데이터의 교집합을 출력하기(INTERSECT)

집합 연산자: 1. 합집합 연산자 : union all, union
2. 교집합 연산자 : intersect
3. 차집합 연산자 : minus

- 수학과 비슷

데이터분석, 딥러닝회사: SQL 쪽지시험, 파이썬 코딩시험
제약회사 데이터 분석가: 수학시험

Ex) 우리반 테이블을 백업하시오!

```
create table emp12_backup
as
select *
from emp12;
```

우리반 테이블에서 김씨 학생들만 가져와서 emp_backup2 테이블을 생성하시오

```
create table emp12_backup2
as
select *
from emp12
where ename like '김%';
```

```
Ex ) select ename, age, telecom
       from emp12
       union all
       select ename, age, telecom
       from emp12_backup2;
       >>> 38행이 나옴

       select ename, age, telecom
            from emp12
            union
            select ename, age, telecom
            from emp12_backup2;
       >>>30행이 나옴
```

※ 설명 : union은 합집합 연산자인데 중복된 데이터를 제거하고 정렬작업도 수행합니다.

```
       select ename, age, telecom
            from emp12
            intersect
            select ename, age, telecom
            from emp12_backup2;
       >>> 8개의 행만 나옴 (교집합)
```

070 집합 연산자로 데이터의 차이를 출력하기(MINUS)

"차집합을 구하는 집합 연산자"

```
Ex ) select ename, age, telecom
       from emp1
       minus
```

071 서브 쿼리 사용하기 1(단일행 서브쿼리)

Ex 1) 사원테이블에서 최대월급을 받는 사원의 이름과 월급을 출력하시오!

```
select ename, max(sal)
  from emp;
```

Ex) select ename, age, telecom
from emp1
minus
select ename, age, telecom
from emp12_backup2;

총 22행이 나옴

※ 설명 : 김씨빼고 다 출력되었습니다.

select ename, max(sal)
from emp;

※ 설명 : 위의 SQL을 수행하려면 서브쿼리을 사용해야 합니다.

Ex 2) JONES의 월급을 출력하시오

```
select sal
    from emp
    where ename = 'JONES';
```

Ex 3) JONES의 월급보다 더 많은 월급을 받는 사원들의 이름과 월급을 출력하시오!

```
select ename, sal
    from emp
    where sal >= (select sal
                  from emp
                  where ename = 'JONES');
```

메인쿼리 서브쿼리

※ 설명 : 쿼리를 두번 각각 실행하지 말고 위와 같이 한번에 수행 하자!
오라클은 서브쿼리 먼저 시행한다.

072 서브 쿼리 사용하기 2(다중 행 서브쿼리)

- **서브쿼리의 종류 3가지:**

1. **단일행 서브 쿼리** : 서브쿼리에서 메인쿼리로 하나의 값이 리턴되는 경우
   연산자 : =, >, < , >=, <-, !=, <>, ^=
2. **다중행 서브 쿼리** : 서브쿼리에서 메인쿼리로 여러개의 값이 리턴되는 경우
   연산자 : in, not in, >all, <all, >any, <any
3. **다중 컬럼 서브쿼리** : 서브쿼리에서 메인쿼리로 여러개의 컬럼값이 리턴되는 경우

Ex) 직업이 SALESMAN인 사원들과 월급이 같은 사원들의 이름과 월급을 출력하시오!

```
select ename, sal
    from emp
    where sal = (select sal
                 from emp
                 where job = 'SALESMAN');
```

>>>오류남!
왜냐하면 직업이 SALESMAN인 사람이 여러명이기 때문.

```
select ename, sal
    from emp
    where sal in (select sal
                  from emp
                  where job = 'SALESMAN');
```

>>>이렇게 해줘야 함

단일행 서브쿼리

```
select ename, sal
    from emp
    where sal >= (select sal
                  from emp
                  where ename = 'JONES');
```

>>> 딱 한개의 값만 리턴됨.

다중행 서브쿼리

```
select ename, sal
    from emp
    where sal in (select sal
                  from emp
                  where job = 'SALESMAN');
```

>>> 두개 이상의 값이 리턴되고 있음.

073 서브 쿼리 사용하기 3(NOT IN)

Ex) 관리자인 사원들의 이름을 출력하시오!
(자기 밑에 직속 부하가 한명이라도 있는 사람)
(이 관리자들의 사원번호는 직속부하 사원들의 mgr입니다)

```
select ename
    from emp
    where empno in (select mgr
                    from emp);
-----------------------
select ename
    from emp
    where empno not in (select mgr
                    from emp);
```

이렇게 하면 결과가 안 나옴.
>>> mgr이 null인 사람이 있어서 (KING)

※ 서브쿼리문에서 not in 연산자 사용할 시 주의할 사항!
서브쿼리에서 null값이 하나라도 리턴되면 결과가 출력되지 않는다.

```
select ename
    from emp
    wher empno in (7839, 7902, 7566, null);
```

이건 왜 값이 잘 나오느냐?

```
select ename
    from emp
    where empno = 7839 or empno = 7902 or empno = 7566 or empno = null;
          True         or   True       or   True       or null
```

이랑 똑같다.

or 은 하나만 true가 있어도 결과가 true가 됨.

※ 설명 : where 절에 조건의 true이면 결과가 잘 출력이 된다.

```
select ename
    from emp
    wher empno not in (7839, 7902, 7566, null);
```

은

```
select ename
    from emp
    where empno != 7839 and empno != 7902 and empno != 7566 and empno != null;
          True          and   True      and   True      and   null
```

이랑 똑같다.

※ null 딱 한 개 때문에 전체가 다 null이 되어버려서 결과가 출력이 안됨.
따라서 서브쿼리문에서 nvl을 써서 null값을 대체해주거나 where절에 ___ is not null 이런식으로 조건을 주어야 합니다.

※ 중요 설명 : 서브쿼리문 사용시 not in 연산자를 사용하면 반드시 서브쿼리에서 메인쿼리로 null값이 리턴되지 않도록 처리를 해줘야 합니다.

074 서브 쿼리 사용하기 4(EXISTS와 NOT EXISTS)

서브쿼리문에서 exist와 not exist를 사용해서 메인쿼리에 있는 데이터 중에 서브쿼리에 존재하는 전제 유무를 파악할 때 사용하는 SQL문법

Ex) emp테이블에 권세원 데이터를 입력을 합니다.

```
Insert into emp(empno, ename, sal, job, deptno)
values (9877, '권세원', 6000, 'ANALYST', 20);

commit;
```

Ex) 우리반 테이블에 있는 학생들 중에 사원 테이블에도 존재하는 학생이 있으면 이름을 출력하시오!

```
select ename
      from emp12 e12
      where exists (select *
                        from emp e
                        where e.ename = e12.ename);
```

※ 설명 : exist문은 main query 컬럼이 서브쿼리 안으로 들어가면서 실행됩니다. 그러면서 메인쿼리의 데이터 중에 서브쿼리에도 같은 데이터가 존재하는지 찾아 봅니다.

11/5 (075-080)

2020년 11월 5일 목요일 오전 9:19

075 서브 쿼리 사용하기 5(HAVING절의 서브 쿼리)

- **select문장에서 서브쿼리를 쓸 수 있는 절**

select 서브쿼리 사용가능 (scalar subquery)
from 서브쿼리 사용가능 (in line view)
where 서브쿼리 사용가능
group by X
having 서브쿼리 사용가능
order by 서브쿼리 사용가능 (scalar subquery)

scalar: 확장

077 서브 쿼리 사용하기 7(SELECT절의 서브 쿼리)

"select절에 서브쿼리를 사용할 수 있는데 select절의 서브쿼리를 scalar subquery 라고 합니다"

Ex) 이름, 월급, 사원테이블의 평균월급을 출력하시오!

```
select ename, sal, avg(sal)
    from emp
    group by ename, sal;
```

이렇게 하면 원하는 값이 안나옴. 평균월급이 아니라 자기 자신의 월급이 나옴

```
select ename, sal, (select avg(sal)
                    from emp) 평균월급
    from emp;
```

078 데이터 입력하기(INSERT)

"테이블에 데이터를 입력하는 SQL문장"

```
insert into emp(empno, ename, sal)
    values(1234, 'jack', 4500);
```

데이터 입력할 컬럼들 기술
위의 컬럼 순서대로 값을 기술

080 데이터 삭제하기(DELETE, TRUNCATE, DROP)

- **오라클에서 데이터를 삭제하는 방법 3가지?**

1. **delete**
2. **truncate**
3. **drop**

Ex) delete from emp
    where empno = 7788;

❖ 설명 : 사원번호가 7788번인 사원을 삭제하겠다.

실수로 다 delete하고 commit까지 눌렀을때:

- 타임머신 기능을 이용해서 과거로 emp테이블 되돌립니다.

1. emp테이블을 flashback이 가능한 상태로 변경한다.
   (타임머신기능을 할 수 있도록 설정한다.)

076 서브 쿼리 사용하기 6(FROM절의 서브 쿼리)

"from 절에도 서브쿼리를 사용할 수 있습니다. from절의 서브쿼리는 in line view 라고 합니다"

Ex) 이름, 월급, 순위를 출력하는데 순위가 월급이 높은 사원 순으로 순위를 부여하시오.

```
select ename, sal, dense_rank() over (order by sal desc) 순위
    from emp;
```

Ex) 위의 결과에서 순위가 4등인 사원만 출력하시오.

```
select ename, sal, dense_rank() over (order by sal desc) 순위
  from emp
  where 순위 = 4;
```

이렇게 하면 에러가 남.

❖ 설명 : 위의 SQL은 에러가 납니다. 왜냐하면 실행 순서 때문입니다.
from절 실행하고 where절을 실행하기 때문에 emp테이블에는 순위라는 집합이 없기 때문에 에러가 납니다.

순위가 4등인 사원의 이름과 월급가 순위를 출력하려면 from절의 서브쿼리를 사용해야 합니다.

```
4 select *
    1 from ( 2 select ename, sal, dense_rank() over (order by sal desc) 순위
             3 from emp)
    where 순위 = 4;
```

❖ 설명 : from 절의 서브쿼리문의 결과가 마치 하나의 테이블처럼 만들어져서 서브쿼리문의 결과 데이터가 메모리에 올라가게 됩니다.
메모리에 올려놓은 데이터 중에 순위가 4등인 데이터를 가져온다.

079 데이터 수정하기(UPDATE)

"데이터를 수정하는 SQL문"

```
Ex ) update emp
     set sal = 0
     where ename='SCOTT';
```

❖ 설명 : SCOTT의 열급을 0으로 수정하는 update문장
rollback;
>>> 방금 실행한 명령을 취소하는 명령

alter table emp enable row movement;

2. 현재시간에서 10분전으로 emp테이블을 되돌린다.
   flashback table emp to timestamp
   (systimestamp - interval '10'minute);

   골든타임이 15분이다!

- **truncate 명령어**

1. 데이터 모두 삭제
2. rollback안됨
3. flashback 안됨
4. 테이블 구조만 남기고 다 지운다.
   운영서버 ---------- 테스트 서버
   고객테이블 고객테이블
   truncate table emp;
   Truncate명령어로 다 날린경우:
   백업받은 데이터로 복구 하는 수 밖에 없다.

- **drop 명령어**

1. 모든 데이터 삭제
2. 테이블 구조까지 다 삭제
3. rollback 안됨
4. 휴지통 기능이 있어서 flashback은 가능합니다.
   show recyclebin; <--- 휴지통 속에 있는 데이터 확인
   flashback table emp to before drop; <---- 휴지통 속에서 복원

11/6 (081-089)

2020년 11월 6일 금요일 오전 9:32

081 데이터 저장 및 취소하기(COMMIT, ROLLBACK)

commit은 지금까지 벼녕한 데이터베이스 작업들(DML 문장)을 데이터베이스에 영구히 반영하겠다는 TCL (Transaction Control Language)문

SQL의 종류

1. Query문 : select문의 6가지절, 조인, 서브쿼리
2. DML문 : insert, update, delete, merge
3. DDL문 : create, alter, drop, truncate, rename
4. DCL문 : grant, revoke
5. TCL문 : commit, rollback, savepoint

rollback은 지금까지 변경한 데이터 베이스 작업들(DML문장)을 취소하는 명령어 마지막으로 commit한 이후의 작업한 DML작업들을 취소한다.

Ex) select count(*) from emp;
↓
delete from emp;
↓
select *
from emp;
---

1. commit;
↓
2. update emp
set sal = 0;
↓
3. commit;
↓
4. delete from emp;
↓
5. rollback;

마지막 commit시점인 3번까지만 rollback이 됨

- 암시적 commit (rollback 못함)

1. 정상 종료 (exit)
2. DDL문 수행 : create, alter, drop, truncate, rename
3. DCL문 수행 : grant, revoke

Ex) SQL > delete from emp;

SQL> create table emp902
(empno number(10),
ename varchar2(10));

>>>delete한 것에 대해 rollback 못함

※ 중요설명 : 오라클은 DML문장을 수행하고서 정상종료 또는 DDL문 수행 또는 DCL문을 수행하지 않았고 명시적으로 COMMIT을 수행하지 않았다면 DML 작업을 취소(rollback)할 수 있습니다. ---> 오라클 SQL

MSSQL은
begin tran <--- 앞에 이 명령어를 수행하지 않고
delete from emp <--- delete문장을 수행했으면 자동 commit된다.

begin tran 안하면 rollback 안된다.
MSSQL은 flashback 안된다.

- 암시적 rollback

1. database시스템이 비정상 종료되었을 때 (정전)

Ex) 은행
저축통장----------> 적금
delete insert
100만원
저축통장에서 적금통장으로 가는 사이에 정전 됐다 :
자동 rollback해줌 ->다시 저축통장으로 들어감

082 데이터 입력, 수정, 삭제 한번에 하기(MERGE)

" 데이터 입력과 수정과 삭제를 한번에 수행하는 명령어이고 SQL튜닝을 위해서 자주 사용되는 SQL"

Ex1) 이름과 부서위치를 출력하시오.

```
select e.ename, d.loc
  from emp e, dept d
  where e.deptno = d.deptno;
```

Ex2) 사원 테이블에 부서위치(loc) 컬럼을 추가한다.

```
alter table emp
     add loc varchar2(10);
```

문자형으로 철자 10자리를 포함시키게끔 추가하겠다.

Ex3) 사원테이블에 추가한 부서위치(loc) 컬럼에 데이터를 해당 사원의 부서위치로 값을 갱신하시오.

```
merge into emp e
using dept d
on (e.deptno = d.deptno) >>> 조인조건
when matched then >>> emp table과 dept table이 매치가 된다면
update set e.loc = d.loc; >>> 업데이트해라
```

※ 설명: merge into 다음에는 변경할 타깃(target) 테이블 명을 작성하고
using 다음에는 소스(source) 테이블명을 작성한다.
on 다음에는 타깃과 소스 테이블간의 연결고리를 작성한다.
when matched then 다음에 변경할 update 문을 작성한다.

083 락(LOCK) 이해하기

"특정 창(session)에 접속한 유저(scott)가 KING의 월급을 9000으로 변경하고 이는 상태에서 다른 창(session)에 접속한 유저(scott)가 KING의 월급을 변경할 수 없도록 막는 기능을 LOCK이라고 합니다."

※ 내가 변경한 데이터를 남이 보려면 내가 commit 해줘야 볼 수 있다.
내가 commit하지 않으면 내가 변경한 데이터를 남이 볼 수 없다.

update를 수행하면 update를 수행하고 있는 행(row)에 락(lock)을 건다.

Ex)

```
update emp
  set sal = 8000
   where ename = 'KING';
```

※ 설명 : 위와 같이 update를 하면 KING의 전체 행에 락(lock)을 겁니다.
다른 창(session)에 접속한 유져가 KING의 데이터를 절대로 갱신할 수 없다.

Ex) 도스창 A

```
rollback;

update emp
  set sal = 8700
  where ename = 'KING';
```

도스창B

```
rollback;

update emp
 set deptno = 30
 where ename = 'KING';
```

동사살행하면 lock 걸릴까? 네 걸립니다. (같은 행이라서)
먼저 실행한 창에서 commit이나 rollup해줘야 풀림

Ex) 도스창 A

```
commit;

update emp
set sal = 9700
where ename 'JAMES';
```

도스창 B

```
commit;

update emp
 set sal = 8000
 where ename = 'ALLEN'
```

※ 설명 : 도스창 A는 JAMES행에 LOCK을 걸었고 도스창 B는 ALLEN의 행에 LOCK을 건거라 서로 충돌되지 않습니다.

오라클에 LOCK이 있어서 사용자들은 항상 일관된 데이터를 볼 수 있습니다.

084 SELECT FOR UPDATE절 이해하기

"보통 lock은 update문을 수행할 때 주로 걸리나 select를 수행할 때는 lock이 걸리지 않는데 select를 수행할 때도 lock을 걸고 싶으면 select for update를 이용하면 됩니다.
내가 어떤 데이터를 보고있는 동안 그 누구도 이 데이터를 갱신하지 못하도록 막고 싶을 때 select for update 문을 사용합니다."

Ex) 코스트코가 밤 9시에 문을 닫는데 9시에 매장에 진열된 상품의 갯수를 파악해서 모자란 상품을 주문해서 채워넣으려고 한다.

9시를 기준으로 지금 매장의 상품들의 상품 갯수를 파악하고 싶다.
항상 10개의 상품이 있어야하는 커피제품이 있다면 지금 남은게 3개면 7개를 주문하려고 한다. 그런데 상품의 갯수가 계속 변경되면 새로 주문을 넣을 때 혼란스러우므로 9시 현재를 기준으로 그 어떤 데이터도 갱신하지 못하게 막아버립니다.

Ex) 도스창A
```
commit;

select ename, sal
    from emp
    where ename = 'BLAKE'
    for udpate;
```

도스창B
```
commit;

update emp
  set sal
  where ename = 'BLAKE';
```

이 경우에도 lock이 걸린다.
도스창A 에서 commit을 눌러줘야 도스창B가 업데이트를 할 수 있다.

085 서브 쿼리를 사용하여 데이터 입력하기

우리가 지금까지 배운 insert문장은 한번에 한건만 입력할 수 있었습니다.

```
insert into emp(empno, ename, sal)
    values(1234, 'aaa', 4000);
```

그런데 서브쿼리를 사용한 insert문장을 이용하면 한번에 여러건의 데이터를 입력할 수 있게 됩니다.

```
select count(*)
  from emp12_backup3; (백업파일 생성)

truncate table emp12; (지우기)

insert into emp12
select *
  from emp12_backup3; (백업)
```

086 서브 쿼리를 사용하여 데이터 수정하기

```
update emp                    <--- 서브쿼리 사용가능
    set sal = 8900            <--- 서브쿼리 사용가능
    where ename = 'SCOTT'; <--- 서브쿼리 사용가능
```

Ex 1) SCOTT보다 더 많은 월급을 받는 사원들의 직업을 SALESMAN으로 변경하시오!
```
update emp
  set job = 'SALESMAN'
  where sal >= (select sal
                from emp
                where ename = 'SCOTT');
```

087 서브 쿼리를 사용하여 데이터 삭제하기

Ex) SCOTT보다 월급을 많이 받는 사원들을 삭제하시오!
```
delete from emp
where sal > ( selet sal
              from emp
              where ename = 'SCOTT');
```

088 서브 쿼리를 사용하여 데이터 합치기

"merge문에 서브쿼리를 사용하기"

Ex 1) 부서번호, 부서번호별 토탈월급을 출력하시오! (세로 출력)

```
select deptno, sum(sal)
    from emp
    group by deptno;
```

Ex 2) 부서테이블에 sumsal이라는 컬럼을 추가하시오!
```
alter table dept
  add sumsal number(10);
```

Ex 3) 위의 dept테이블에 추가한 sumsal 컬럼에 해당 부서번호의 토탈월급으로 값을 갱신하시오!
```
merge into dept d
  using (select deptno, sum(sal) 토탈
         from emp
         group by deptno) e
  on (e.deptno = d.deptno)
  when matched then
  update set d.sumsal = e.토탈;
```

089 계층형 질의문으로 서열을 주고 데이터 출력하기 1

카카오 면접 때 받았던 질문:

1. 1부터 10까지의 합은 무엇인가요? SQL로 수행할 수 있다.

Ex) select 1+2+3+4+5+6+7+8+9+10
    from dual;

2. 1부터 100까지의 합을 구하세요! 계층형 질의문을 사용하면 쉽게 할 수 있습니다.

```
select level
    from dual <<< 결과를 보기 위한 가상의 테이블
    connect by level  <= 100;
```

※ 설명 : 1번부터 connect by절의 level 다음에 나오는 숫자만큼 숫자가 증가해서 출력이 된다.

알고리즘 문제 1
1부터 10까지의 합을 구하시오!
```
select sum(level)
  from dual
  connect by level  <= 100;
```

```
        group by deptno) e
on (e.deptno = d.deptno)
when matched then
update set d.sumsal = e.토탈;
```

```
select sum(level)
    from dual
    connect by level   <= 100;
```

11/9 (090-093)

2020년 11월 9일 월요일 오전 9:38

090 계층형 질의문으로 서열을 주고 데이터 출력하기 2

"순위와 서열을 출력하는 SQL 문"

- 계층형 질의절은 WHERE절 다음에 기술하며, from절이 수행된 후 수행된다.
- start with 절과 connect by절로 구성되며, start with 절이 수행된 후 connect by절이 수행된다. start with절은 생략이 가능하다.

절	설명
start with 절	루트 노드를 생성하며 한번만 수행한다.
connect by 절	루트 노드의 하위 노드를 생성하며 조회 결과가 없을 때 까지 반복한다.

Ex 1) 1부터 10까지의 숫자를 출력하시오.

```
select level <--- connect by 절을 써야지만 출력되는 절입니다.
  from dual <---결과를 보기 위한 가상의 테이블
  connect by level <=10; <--- level이 10보다 작거나 같을때까지 계속 해서 연결하겠다.
```

Ex 2) 사원 테이블의 사원 순서를 출력하시오!
(KING이 사장이고 KING의 부하직원들이 있을거고 그리고 그 부하직원들 밑에 또 부하직원들이 있을 것이고 이렇게 서열을 출력하시오!)

```
select level, empno, ename, mgr
    from emp
    start with ename = 'KING' <--- KING을 root로 해서 시작하겠다.
    connect by prior empno = mgr;
                  사원번호   관리자의 사원번호
    connect by 절과 prior절은 짝꿍
```

❖ 설명 : select절에 level이라는 컬럼은 emp테이블에는 없는 컬럼입니다. 그런데 출력될 수 있었던 것은 connect by절을 썼기 때문입니다.

091 계층형 질의문으로 서열을 주고 데이터 출력하기 3

계층형 질의문과 짝꿍 함수인 sys_connect_by_path를 이용하면 listagg를 사용한것처럼 가로로 결과를 출력할 수 있습니다.

```
Ex ) select ename, sys_connect_by_path(ename, '/') as path
     from emp
     start with ename = 'KING'
     connect by prior empno = mgr;
```

결과 :

```
KING   /KING
JONES /KING/JONES
SCOTT /KING/JONES/SCOTT
ADAMS /KING/JONES/SCOTT/ADAMS
FORD   /KING/JONES/FORD
SMITH /KING/JONES/FORD/SMITH
BLAKE  /KING/BLAKE
ALLEN /KING/BLAKE/ALLEN
WARD /KING/BLAKE/WARD
MARTIN /KING/BLAKE/MARTIN
TURNER /KING/BLAKE/TURNER
JAMES /KING/BLAKE/JAMES
CLARK /KING/CLARK
MILLER /KING/CLARK/MILLER
```

❖ 설명 : 계층적 질의문 작성할 때 반드시 알아야 하는 두가지 짝꿍 키워드

1. order by 절의 siblings
2. 서열을 가로로 출력하는 sys_connect_by_path 함수

092 계층형 질의문으로 서열을 주고 데이터 출력하기 4

계층형 질의문에서도 조인문장을 같이 쓸 수 있습니다.

```
Ex ) 이름, 서열, 월급과 부서위치를 출력하시오!
     select rpad(' ', level*2)|| e.ename, level, e.sal, d.loc
          from emp e, dept d
          where e.deptno = d.deptno <---조인조건 (연결고리)
          start with ename = 'KING'
          connect by prior empno = mgr;
```

```
select regexp_count(lower(win_text), 'elsa') as cnt
     from winter_kingdom;
```

❖ 설명 : win_text를 전부 소문자로 변경하고 regexp_count를 이용해서 스크립트 한행 한행을 다 살펴봐서 elsa라는 단어가 포함되어져 있으면 카운트 된다.

```
select sum(regexp_count(lower(win_text), 'elsa')) as cnt
from winter_kingdom;
>>> 329
```

093 일반 테이블 생성하기(CREATE TABLE)

DDL문 : (Data Definition Language : create, alter, drop, truncate, rename)

Ex) 겨울왕국 대본을 오라클에 입력하고 elsa가 많이 나오는지 anna가 많이 나오는지 또는 긍정단어가 많은지 부정단어가 많은지 등을 SQL로 검색하려면 DDL문을 이용해서 테이블을 생성 할 수 있어야한다.

```
Ex ) 테이블 생성 스크립트
     create table emp500    --> 테이블명
     (empno number(10),     --> 데이터 유형(길이)
     ename varchar2(20),          1. 문자형 : varchar2, char
     sal      number(10));        2. 숫자형 : number
                                  3. 날짜형 : date
          number(10) --> 숫자 10자리를 허용하겠다.
          varchar2(20)--> 영어철자 20개를 허용하겠다.
```

```
select sum(regexp_count(lower(win_text), 'elsa')) as cnt
from winter_kingdom;
>>> 329

select sum(regexp_count(lower(win_text), 'elsa')) as cnt
from winter_kingdom;
>>> 685
```

❖ 설명 : 카운트된 숫자들을 sum함수를 이용해서 다 더한다.

❖ 설명 : 위의 SQL이 유용한 경우?

1. 뉴스기사를 분석해서 회사 주식에 영향을 주는지 확인 가능
   (Ex : 미래에셋 로보어드바이저)

2. 케냐의 사례중에 은행 고객 대출 한도를 정할 때 대출 신청자의 SNS 글을 분석해서 긍정적인 단어가 많은지 부정적인 단어가 많은지 분석해서 대출여부를 결정한다.

number(10) --> 숫자 10자리를 허용하겠다.
varchar2(20)--> 영어철자 20개를 허용하겠다.

- datatype의 유형

데이터 유형	설명
char	고정길이 문자 데이터유형이고 최대 길이가 2000 입니다
varchar2	가변길이 문제 데이터유형이고 최대길이가 4000입니다
long	가변길이 문자 데이터유형이고 최대 2GB까지 데이터 허용
clob	문자 데이터 유형이고 최대 4GB까지 문자 데이터 허용
blob	바이너리 데이터 유형이고 최대 4GB (사진, 동영상)
number	숫자 데이터 유형이고 최대 38까지 허용
date	날짜 데이터 ㅠ형이고 기원전 4712년 1월 1일부터 기원후 9999년 12월 31일까지 날짜를 허용

number(10,2) : 숫자 데이터 10자리를 허용하는데 그중에 소수점 2자리를 허용하겠다.
Ex) 3500.23

- long 사용하기

```
create table profile2
    (ename varchar2(20),
    self_intro long);

insert into profile2(ename, self_intro)
  values('김인호' ,'어렸을때 부터 우리집은 가난했었습니다. 그리고 어머니는 짜장면이 싫다고 하셨습니다. 야히 야히야');
```

11/10 (094-097)

2020년 11월 10일 화요일 오전 9:42

- 복습

1. select문의 6가지 절:
   select
   from
   where
   group by
   having
   order by
2. 함수 : 단일행 함수 : 문자, 숫자, 날짜, 변환, 일반
   복수행 함수 : max, min, avg, sum, count
3. 조인문장 : 오라클 조인문장 : equi join
   non equi join
   outer join
   self join
   1999 Ansi 조인문장 : on 절을 사용한 조인
   using절을 사용한 조인
   natural join
   left/right/full outer 조인
   cross 조인
4. 서브쿼리 : single row subquery
   multiple row subquery
   multiple column subquery
5. 집합연산자 : union all
   union
   intersect
   minus
6. 계층형 질의문 : 서열을 보여주는 sql문

```
select level
    from emp
    start with ename = 'KING'
    connect by prior empno = mgr;
                          ↑      ↑
                       부모키    자식키
```

7. DML문장 : insert, update, delete, merge
8. DDL문장 : create, alter, drop, truncate, rename

- 데이터분석가 : SQL
  파이썬 (딥러닝)
  R (시각화, 통계)
  하둡 -> SQL -> Hive -> Java
- 딥러닝 개발자(연구원)

현업에서 계층형 질의문을 활용했을 때의 장점?
자바
c
기타 프로그래밍 언어

길게 작성해야하는 코드 ---------->SQL하나로 끝낼 수 있다.

- 치환변수 (&) 사용법
  SQL 문장을 수행할 때 매번 검색해야되는 데이터 값이 다른데
  SQL 문장이 같을 때 용이하게 하는 오라클 SQL문법
  (반복해서 결과를 봐야할 때 유용)

```
Ex ) select empno, ename, sal
       from emp
       where empno =&사원번호;
```

- 구구단 1단부터 9단까지 출력하기 위해 알아야 할 SQL
  Ex 1) 숫자 1과 2를 출력하는 SQL문을 작성하시오!

```
select level
       from dual
       connect by level <= 2
```

Ex 2) 위의 결과를 from절의 서브쿼리로 만들고 emp 테이블과 조인하시오

```
select empno, ename, sal
       from emp e, (select level
                    from dual
                    connect by level <= 2);
```

```
EMPNO ENAME      SAL
------------------------------
7839  KING       5000
7698  BLAKE      2850
7782  CLARK      2450
7566  JONES      2975
7654  MARTIN     1250
7499  ALLEN      1600
7844  TURNER     1500
7900  JAMES       950
7521  WARD       1250
7902  FORD       3000
7369  SMITH       800
7788  SCOTT      3000
7876  ADAMS      1100
7934  MILLER     1300
7839  KING       5000
7698  BLAKE      2850
7782  CLARK      2450
7566  JONES      2975
7654  MARTIN     1250
7499  ALLEN      1600
7844  TURNER     1500
7900  JAMES       950
7521  WARD       1250
7902  FORD       3000
7369  SMITH       800
7788  SCOTT      3000
7876  ADAMS      1100
7934  MILLER     1300
```

※ 설명 : emp테이블의 모든 데이터가 숫자 1과 조인해서 14개 출력하고 emp 테이블의 모든 데이터가 숫자 2와 조인해서 14개 출력해서 총 28개가 출력되었다.

Ex 3) 숫자를 1부터 9까지 출력하는 SQL문장을 출력하시오.

```
select level
   from dual
   connect by level <=9;
```

Ex 4) 위의 숫자를 1부터 9까지 출력하는 SQL문을 아래와 같이 FROM절의 서브쿼리로 2개로 만드시오.

```
select *
       from  (select level as num1
                    from dual
                    connect by level <=9) a,
             ( select level as num2
             from dual
             connect by level <=9) b;
```

094 임시 테이블 생성하기(CREATE TEMPORAY TABLE)

"데이터를 영구히 database에 저장하는게 아니라 임시로 저장하는 테이블"

데이터 중에서 영구히 저장할 필요는 없고 지금 잠깐 테스트를 위해서 볼 데이터라든가 아니면 지금 현자만 필요하고 나중에는 필요하지 않는 데이터가 있는데 그 데이터를 잠깐 저장할 때 사용하는 테이블이 임시 테이블 입니다.

- **임시테이블 종류 2가지**

1. **on commit delete row 옵션 : 데이터를 commit할때까지만 보관**
2. **on commit preserve rows 옵션 : 데이터를 접속한 유저가 로그아웃할때까지만 보관**

```
Ex ) create global temporary table emp700
       (empno number(10),
       ename varchar 2(10),
       sal number(10))
       on commit delete rows;

       insert into emp700
       select empno, ename, sal
          from emp;

       commit;
```

하면 데이터 사라진 걸 볼 수 있음

```
Ex 2)
create global temporary table emp800
       (empno number(10),
       ename varchar 2(10),
       sal number(10))
       on commit preserve rows;

       insert into emp800
       select empno, ename, sal
          from emp;

       commit;
```

해도 데이터 그대로 남아있음.
그러나 exit하고 다시 들어가면 데이터가 사라짐

095 복잡한 쿼리를 단순하게 하기 1(VIEW)

"테이블은 아니고 데이터를 바라보는 쿼리문을 하나의 테이블처럼 하나의 object(객체)로 생성한 것을 말한다"

```
Ex ) create table emp708
       as
       select empno, ename, sal, deptno
       from emp;

       select *
            from emp708;
```

※ 설명 : as 다음에 나오는 쿼리문의 결과대로 emp708테이블이 생성된다.
emp테이블과는 별개의 또 다른 테이블이다.

- emp로 시작하는 내가 만든 테이블이 뭐가 있는지 확인하는 방법

```
select table_name
from user_tablees
where table_name like 'EMP%';
```

↓
빈드시 대문자로 작성

- view 생성

```
create view emp801
as
   select empno, ename, sal, deptno
      from emp;
```

※ 설명 : view는 테이블과는 다르게 별도로 데이터를 저장하고 있지 않는다. 그냥 그 테이블을 바라보는 쿼리문입니다.

```
update emp801
set sal = 0
where ename='SCOTT';
```

" view를 업데이트 하면 실제 테이블도 업데이트 됩니다."

096 복잡한 쿼리를 단순하게 하기 2(VIEW)

- view가 중요한 이유:

회사의 데이터 공개하면 안되는 데이터가 있다. 민감한 데이터

Ex) 라이나생명 데이터 분석을 하러 갔는데 라이나 생명에서 일하는 직원들의 데이터가 있는데 직원 데이터 중에 월급은 라이나 생명에서 공개하지 않고 나머지 데이터만 공개를 하고 얼마든지 조회하고 업데이트 가능하게 하고 싶다면 어떻게 분석가에게 데이터를 줘야되는가?

답 : 월급만 빼고 view로 생성해주면 된다. (월급 빼고 다 업데이트 가능하기 때문)

```
create view emp809
as
       select empno. ename, job, hiredate, mgr, deptno
            from emp;
```

- 뷰의 장점? 1. 민감한 컬럼을 감춰서 데이터를 제공할 수 있다.
  2. 복잡한 쿼리문을 단순하게 만들 수 있다.
- view의 종류 2가지

	□ 단순 view	복합 view
테이블의 개수	1개	2개이상
그룹함수	포함안됨	포함됨
DML 여부	가능	불가능할 수도 있다.

- view 삭제

```
drop view emp809;
```

097 데이터 검색 속도를 높이기(INDEX)

" 오라클 데이터베이스의 객체(object)의 종류 5가지

1. **table : 데이터를 저장하는 기본 저장소**
2. **view : 데이터를 바로보는 쿼리문**
3. **index : 검색속도를 높이기 위한 db object**
4. **sequence : 순서대로 번호를 생성하는 db object**
5. **synonym : 테이블명에 대한 또 다른 이름**

index : SQL 튜닝을 위한 반드시 알아야하는 db object
↓
검색속도를 높이는 기술

※ 설명 : 인덱스는 책으로 치면 책 앞에 나오는 목차가 인덱스입니다.
책이 테이블이면 인덱스는 책의 목차입니다.
목차가 없는 상태에서 책의 내용중에 어떤내용을 찾는다며 처음부터 끝까지 다 스캔해야 할 것입니다.

Ex 1)이름이 SCOTT인 사원의 이름과 월급을 출력하시오.

```
select ename, sal
       from emp
       where ename = 'SCOTT';
```

※ 설명 : ename에 인덱스가 없기 때문에 SCOTT을 emp테이블에서 찾을 때 처음부터 끝가지 full table scan을 했을 것입니다.

Ex 2) emp 테이블의 ename에 인덱스를 생성하시오!

```
create index emp_ename <<< 인덱스 이름
    on emp(ename);
         ↑
       테이블이름(컬럼명)

select ename, sal
    from emp
    where ename = 'SCOTT'
```

※ 설명 : **위와 같이 인덱스를 생성하면 왜 검색이 빠른것인가?**

책으로 치면 앞에 목차가 만들어져서 입니다. 그래서 목차를 먼저 검색하고 책을 검색하면 훨씬 빠르게 원하는 데이터를 검색할 수 있기 때문입니다.

- 인덱스(목차)의 구조
  소제목 + 페이지 번호
  소제목이 ㄱㄴㄷㄹ 순서대로 구성되어 있다.

  위와 같이 ename에 목차(인덱스)를 만들면 ename을 이용해서 만든 인덱스의 구조는 컬럼명 + rowid로 구성되어 있고 그리고 컬럼명은 ABCD순서대로 정렬되어 인덱스(목차)를 구성합니다.
  rowid는 그 행을 대표하는 주소입니다.

  ```
  select rowid, ename, sal
      from emp;
  ```

- emp_ename의 인덱스의 구조를 보는 쿼리문

  ```
  select ename, rowid
      from emp
      where ename > ' '; <<< where 절을 써줘야 order by 안써줘도 이름이
                            ABCD순으로 나옴

  select ename, rowid
      from emp; <<< 이렇게 하면 ABCD순서로 안나옴
  ```

11/11 (098-104)

2020년 11월 11일 수요일 오전 9:26

- 어제 배운 내용 복습

1. view 생성하는 방법과 view를 사용해야하는 이유
2. 인덱스가 무엇인지? 인덱스를 생성하는 방법? 인덱스의 구조

SQL문제 5문제 손코딩 쪽지시험
파이썬 노트북을 이용해서 파이썬 알고리즘 코딩 2~3문제

database를 배울 때 인덱스(Index)를 알아야 하는 이유?

인덱스(Index)를 알아야 빠르게 데이터를 검색할 수 있습니다.
인덱스는 책의 목차와 같고 책의 내용을 테이블 입니다.

20분 ---> 0.01초: 쉬운 SQL튜닝
0.1초 ---> 0.01초: 어려운 SQL튜닝 ---->

1. 인덱스의 구조를 이해하고 활용을 잘 할 수 있어야 함
2. 중급 SQL튜너, 고급 SQL튜너
   알고리즘 문제 SQL처럼 생각을 많이 하고 시간이 많이 걸리는 SQL을 스스로 했느냐

인덱스의 구조: 컬럼명 + rowid (row의 물리적 주소)
컬럼명이 오름차순으로 정렬이 되어있다.

Ex) emp테이블에 ename 인덱스를 걸겠다.

```
create index, emp, ename
      on emp(ename);
```

❖ 설명 : emp테이블의 ename에 인덱스가 있으므로 ename의 인덱스인 emp_ename을 통해서 테이블의 데이터를 검색하게 된다. 만약에 목차(index)가 없다면 full table scan을 해야합니다. 만약에 우리가 대용량 데이터가 있는 환경에 가서 SQL을 수행하는데 너무 느리다면 반드시 실행계획을 확인해서 full table scan을 했는지 index scan을 했는지 확인을 해야 합니다.

```
explain plan for
select ename, sal
      from emp
      where ename = 'SCOTT';

select * from table (dbms_xplan.display);
PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Plan hash value: 3956160932

-----------------------------------------------------------------------
| Id  | Operation         | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------
|   0 | SELECT STATEMENT  |      |     1 |    10 |     3   (0)| 00:00:01 |
|*  1 |  TABLE ACCESS FULL| EMP  |     1 |    10 |     3   (0)| 00:00:01 |
-----------------------------------------------------------------------
```

❖ 설명 : 위와 같이 실행계획이 full table scan으로 나오면 SCOTT을 검색하기 위해서 emp 테이블을 처음부터 끝까지 스캔했다는 뜻입니다.

❖ index 를 만들고 select * from table (dbms_xplan.display); 하면

```
-----------------------------------------------------------------------------------------------
| Id  | Operation                           | Name       | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                    |            |     1 |    10 |     2   (0)| 00:00:01 |
|   1 |  TABLE ACCESS BY INDEX ROWID BATCHED| EMP        |     1 |    10 |     2   (0)| 00:00:01 |
|*  2 |   INDEX RANGE SCAN                  | EMP_ENAME  |     1 |       |     1   (0)| 00:00:01 |
-----------------------------------------------------------------------------------------------
```

이렇게 나옴

근데 인덱스가 있는데도 안나오면

```
select /*+ index(emp, emp_ename) */ ename, sal
      from emp
      where ename = 'SCOTT';
```

:힌트

❖ 설명 : emp 테이블 emp_ename에 인덱스를 통해서 검색을 해라~ 하고 힌트를 줘서 옵티마이저에게 명령을 내린다

이름이 BLAKE인 사원의 이름과 월급을 출력하시오!

```
select ename, sal
      from emp
      where ename = 'BLAKE';
```

이 SQL이 어떻게 인덱스를 통해서 테이블을 조종하는가?

```
인덱스                              테이블
select ename, rowid                 select rowid, ename, sal
      from emp                          from emp;
      where ename > ' ';
```

- 인덱스의 구조를 전부 읽어오는 방법

1. 문자 > ' '
2. 숫자 >= @
3. 날짜 <= to_date ('9999/12/31, 'RR/MM/DD')

위의 조건이 where절에 있어줘야 인덱스 전체를 다 읽어올 수 있다.

인덱스의 구조를 알면 SQL튜닝을 할 수가 있는데 그 방법중에 하나가 order by절을 사용하지 않고 결과를 볼 수 있다는 것입니다.
order by 절을 남발하면 검색성능이 느려집니다.

- order by절을 쓰지 않고 인덱스를 통해서 정렬하는 방법

Ex)

1. order by 절을 사용했을 때

```
select ename, sal
      from emp
      order by sal asc;
```

2. 인덱스를 이용하는 방법

```
select ename, sal
      from emp
      where sal >= 0;

select /*+ index_asc(emp emp_sal) */ename, sal
      from emp
      where sal >= 0;
```

❖ 설명 : 인덱스를 통해서 정렬된 데이터를 볼려면 where절에 반드시 채팅 정렬을 검색하는 조건이 있어야 하고 그리고 힌트도 주면 확실하게 결과를 볼 수 있습니다.

힌트 : /*+ index asc(테이블명 인덱스이름) */
      /*+ index desc(테이블명 인덱스이름) */

- 인덱스를 활용한 튜닝 방법

Ex) 튜닝전 :

```
explain plan for
select /*+ index(emp emp_sal) */ ename, sal
      from emp
      where sal * 12 = 36000;
```

❖ 설명 : emp 테이블에 emp_sal인덱스를 엑세스해라 라고 힌트를 줬는데도 불구하고 실행계획이 full table scan을 했습니다.
이유 ? 왜냐하면 where절에 인덱스 컬럼이 가공이 되었기 때문입니다.

```
select * from table(dbms_xplan.display);
```

튜닝후 :

```
explain plan for
select /*+ index(emp emp_sal) */ ename, sal
      from emp
      where sal = 36000/12;
```

- 내가 생성한 인덱스 목록을 확인하는 방법

```
      select index_name
```

```
from user_indexes
where index_name like 'EMP%';
```

- 인덱스 삭제하는 방법
  drop index 인덱스이름

  Ex) drop index EMP_SAL;

098 절대로 중복되지 않는 번호 만들기(SEQUENE)

"번호를 중복되지 않게 순서대로 생성하는 번호 생성기"

```
Ex ) create sequence seq1
     start with 1
     maxvalue 100;

     select seq1.nextval
     from dual;
```

❖ 설명 : 시퀀스를 이용하면 좋은점이 무엇인가?
번호를 중복되지 않게 일관되게 테이블에 입력할 수 있다.
번호가 중복되면 안되는 컬럼은? 주식 테이블 매매번호,
사원 테이블의 사원번호 등

Ex 1) 시퀀스를 emp2라고 해서 만드시오! 1번부터 시작해서 1000번까지

```
create sequence seq2
   start with 1
   maxvalue 1000;
```

Ex 2) 아래의 컬럼을 담는 테이블을 emp534라는 이름으로 생성하시오!

```
empno
ename
sal

create table emp534
   (empno number(10),
   ename varchar2(20),
   sal number(10));
```

Ex 3) 예제1번에서 만든 seq2를 이용해서 emp534이 empno에 일관된 번호가 입력되게 하시오.

```
insert into emp534
      value(seq2.nextval, 'SCOTT', 3000);
select *
      from emp534;
```

insert문장 실행 시킬 때마다 empno의 숫자가 1씩 커지는 것을 볼 수 있음

- 시퀀스의 현재값 확인하는 방법

```
select seq2.currval
      from dual;
```

- 내가 가지고 있는 시퀀스 확인하는 방법

```
select sequence.name
      from user_sequences;
```

- 시퀀스 삭제하는 방법
  drop sequence seq2;

101 실수로 지운 데이터 복구하기 3(FLASHBACK DROP)

"테이블을 drop했을 경우에도 복구할 수 있는데 10g 버전 이후부터는 drop한 테이블이 휴지통에 입력됩니다. 그래서 휴지통에서 끄집어내기만 하면 복구 될 수 있습니다."

Ex) drop table dept;

```
show recyclebin;
```

099 실수로 지운 데이터 복구하기10 1(FLASHBACK QUERY)

오라클은 10g 버전부터 타임머신 기능이 생겼습니다.
그래서 타임머신 기능을 이용해서 과거의 데이터를 확인할 수 있고
테이블을 과거로 되돌릴 수 있습니다.
flashback query는 과거의 데이터를 확인하는 기능입니다.

```
Ex ) delete from emp;
     commit;

     select *
          from emp as of timestamp ( systimestamp - interval '10' minute);
```

설명 : 10분 전에 emp 테이블의 상태를 확인 할 수 있습니다.

```
create table emp_backup_20201111
as
select *
     from emp as of timestamp ( systimestamp - interval '10' minute);
```

100 실수로 지운 데이터 복구하기 2(FLASHBACK TABLE)

"특정 테이블을 과거로 완전히 되돌려 버리는 기능"

```
Ex ) delete from dept;
     commit;
```

1. dept 테이블이 flashback 될 수 있도록 설정한다.
   alter table dept enable row movement;

2. dept 테이블을 과거로 되돌린다.
   flashback table dept to timestamp
   (systimestamp - interval '5' minute);

3. dept테이블을 조회합니다.
   select * from dept;

❖ 설명 : 과거로 되돌리지 못하는 경우
1. sys유저에서 작업했을 때
2. delete하고 commit한 이후에 DDL 명령어를 수행 했을 때

- 과거로 되돌릴 수 있는 골든 타임을 확인하는 방법
  show parameter undo_retention

  undo_retention integer 900 <--- 15분

- flashback table 명령어를 수행해서 과거로 되돌렸으면 반드시 commit을 하여야 합니다.

102 실수로 지운 데이터 복구하기 4(FLASHBACK VERSION QUERY)

특정 테이블이 과거로 부터 현재까지 어떻게 변경이 되어있는지 그 이력정보를 확인하고자 할 때 사용하는 쿼리문

Ex)

1. 현재시간을 확인합니다.
   select systimestamp from dual;
   20/11/11 15:50:18.841000000 +09:00

Ex) drop table dept;

show recyclebin;

select * from dept;
(또는
select *
 from user_recyclebin;)

flashback table dept to before drop;

❖ 설명 : 휴지통만 비우지 않으면 계속해서 휴지통에 있기 때문에 복구 할 수 있습니다.

- 휴지통비우기
  purge recyclebin;

- **테이블 삭제할 때 휴지통에 넣지 않고 삭제하는 방법**
  drop table emp800 purge;

103 실수로 지운 데이터 복구하기 5(FLASHBACK TRANSACTION QUERY)

실습이 잘 안되고 dba영역에 가까워서 따로 실습하지는 않겠습니다.

그동안 작업했던 DML 문장을 확인해 볼 수 있습니다.
어떤 UPDATE의 문장을 수행했고 어떤 DELETE문장을 수행했고 어떤 INSERT문장을 수행했는지 그 문장들을 반대로 수행하는 DML문장이 출력됩니다.

1. 현재시간을 확인합니다.
   select systimestamp from dual;
   20/11/11 15:50:18.841000000 +09:00

2. KING의 이름과 월급과 부서번호를 조회합니다.
   select ename, sal, deptno
   from emp
   where ename = 'KING';

3. KING의 월급을 8000으로 변경하고 commit합니다.
   update emp
   set sal = 8000
   where ename = 'KING';

   commit;

4. KING의 부서번호를 20번으로 변경하고 commit 합니다.
   update emp
   set deptno = 20
   where ename = 'KING';

   commit;

5. 그동안 KING의 데이터가 어떻게 변경되어 왔는지 그 이력정보를 확인하시오!
   select ename, sal, deptno, versions_starttime, versions_endtime,
   versions_operation
   from emp
   versions between timestamp
   to_timestamp('20/11/11 15:50:10','RR/MM/DD HH24/MI/SS')
   and maxvalue
   where ename='KING'
   order by versions_starttime;

```
KING    8000    10      20/11/11 15:53:17.000000000 20/11/11 15:54:07.000000000  U
KING    8000    20      20/11/11 15:54:07.000000000                              U
KING    5000    10      20/11/11 15:53:17.000000000
```

6. 위의 시간중에 하나를 확인해서 그 시간대에 emp테이블의 상태를 확인하시오!
   select *
   from emp as of timestamp
   to_timestamp('20/11/11 15:54:07' , 'RR/MM/DD HH24/MI/SS');

7. 20/11/11 15:54:07 이 시간으로 emp테이블을 복구하시오!
   alter table emp enable row movement;
   flashback table emp to timestamp
   to_timestamp('20/11/11 15:54:07' , 'RR/MM/DD HH24/MI/SS');

❖ 설명 : 일단 과거로 가면 현재로 돌아올 수 없다. 그래서 시간을 정확하게 확인해서 신중하게 되돌려야 합니다.

104 데이터의 품질 높이기 1(PRIMARY KEY)

데이터 분석을 하다보면 제일로 많은 시간을 할애하는 작업이 데이터 전처리입니다. 품질이 높은 데이터를 처음부터 입력받게끔 강제화 하면 데이터 전처리에 많은 시간을 들이지 않아도 됩니다.
그래서 데이터 품질을 높이기 위한 한 방법으로 제약을 사용합니다. 처음부터 테이블에 데이터를 입력받을 때부터 엄격한 기준으로 데이터를 입력받게끔 화려한 제약을 이용하면 됩니다.

데이터 분석가, 딥러닝 개발자에게 모두 중요

- 제약의 종류

1. primary key : 중복된 데이터와 null값을 허용하지 않게 하는 제약
2. unique : 중복된 데이터를 허용하지 않게 하는 제약
3. not null : null값을 허용하지 않게 하는 제약
4. check : 특정 데이터 외에 다른 데이터는 입력되지 못하게 하는 제약
5. foreign key : 참조하는 컬럼에 거는 제약

"primary key 제약을 생성해서 테이블 생성하기"
create table emp307
(empno number(10) primary key, ename varchar2(20));

❖ 설명 : 위와 같이 테이블을 생성하면 empno에는 앞으로 중복된 데이터와 null값이 입력되지 않습니다.

```
insert into emp307 values(1111, 'scott');
insert into emp307 values(2222, 'smith');
insert into emp307 values(1111,'allen'); <<< 중복된 data 입력 불가
insert into emp307 values(null, 'jones'); <<< null 값도 입력불가
```

11/12 (105-121)

2020년 11월 12일 목요일 오전 9:49

105 데이터의 품질 높이기 2(UNIQUE)

"중복된 데이터를 허용하지 않게 하는 제약"

데이터의 품질을 높이기 위해서 중복된 데이터가 테이블에 입력되지 못하게 하는 제약입니다.

Ex) create table emp507
(empno number(10),
ename varchar2(10) unique);

----> ename컬럼에 중복 데이터 입력 불가

insert into emp507 values(1111, 'scott');
insert into emp507 values(2222, 'scott')' ---> 입력불가
두번째 줄 입력했을 때:

명령의 6 행에서 시작하는 중 오류 발생 -
insert into emp507 values(2222, 'scott')
오류 보고 -
ORA-00001: 무결성 제약 조건(SCOTT.SYS_C007461)에 위배됩니다
라고 나옴

- 제약 삭제

1. 삭제한 제약 이름을 확인한다.

select table_name, contraint_name
from user_constraints <--- 제약을 확인하는 데이터 사전
where table_name = 'EMP507'; ---> 반드시 대문자로 작성

2. EMP507의 UNIQUE제약을 삭제한다.

alter table emp507 (테이블명)
drop constraint SYS_C007461; (제약이름)

3. 중복된 데이터가 입력이 잘 되는지 확인하시오!

insert into emp507 values(2222, 'scott');

설명 : 위와 같이 제약이름이 SYS_C007315 라고 나와서 이름만 봐서는 이 제약이 어떤 제약인지 확인하기가 어려우니 제약을 처음 만들때부터 이름을 의미있게 부여해서 만들어주면 관리가 쉽다.

4. 제약이름을 주고 테이블을 생성

create table emp508
(empno number(10),
ename varchar2(10) constraint emp508_ename_un unique);
테이블명_컬럼명_제약이름축약

5. EMP508 테이블에 ename에 걸린 unique제약을 확인한다.

select table_name, constraint_name
from user_constraints
where table_name = 'EMP508';

- 제약을 생성하는 방법 2가지

1. 테이블을 생성할 때
2. 이미 만들어 놓은 테이블에 제약을 추가

- 이미 만들어 놓은 테이블에 제약을 추가하는 방법

1. emp 테이블의 ename에 unique제약을 거시오!

alter table emp
add constraint emp_ename_un unique(ename);
추가 제약이름 이름

※ 설명 : 기존의 테이블에 중복된 데이터가 있다면 위의 명령어는 실행되지 않는다. 예를들어 KING이라는 이름이 2명이 있으면 제약이 실행되지 않는다.

DATABASE에서 제약을 사용해야하는 이유가 무엇입니까?
답변 : 데이터의 품질을 높이기 위해서 입니다.

106 데이터의 품질 높이기 3(NOT NULL)
"null값을 입력하지 못하게 막는 제약"

예: 성별: 남자, 여자
남, 여

Ex) 1. 테이블 생성할 때 제약거는 방법

create table emp707
(empno number(10) constraint emp707_empno_nn not null,
ename varchar2(20));

insert into emp707 values(1111, 'smith');
insert into emp707 values(null, 'scott'); <<< 입력불가

2. 만들어진 테이블에 제약거는 방법

alter table emp707
modify ename constraint emp_empno_nn not null;

107 데이터의 품질 높이기 4(CHECK)
" 지정된 데이터만 입력되고 다른 데이터는 입력되지 못하게 막는 제약"

Ex) 1. 테이블 생성할 때 제약을 거는 방법

create table emp745
(ename varchar2(10),
loc varchar2(10) constraint emp745_loc_ck
check(loc in ('DALLAS', 'CHICAGO', 'BOSTON')));
대/소문자 구분해야 함

※ 설명 : loc에는 DALLAS, CHICAGO, BOSTON 외에는 데이터가 입력되거나 수정되지 않습니다.

2. 만들어진 테이블에 제약을 거는 방법

alter table dept
add constraint dept_loc_ck
check(loc in ('NEW YORK', 'DALLAS', 'CHICAGO', 'BOSTON'));
위의 조건의 데이터만 입력되는지 체크하겠다.
insert into dept (deptno, loc, dname)
values(50, 'SEOUL', 'RESEARCH');

108 데이터의 품질 높이기 5(FOREIGN KEY)

"참조하는 컬럼에 거는 제약"

※ 설명 : dept 테이블에 primary key 제약을 걸고 emp 테이블에 foreign key 제약을 걸면서 emp테이블의 deptno가 dept테이블의 deptno를 참조하겠다라고 하게 되면 앞으로 emp 테이블의 deptno에 부서번호를 입력할 때 dept테이블에 있는 deptno인 10, 20, 30, 40번 데이터만 입력될 수 있습니다. 그리고 dept테이블의 deptno의 데이터를 지우려고 하면 emp 테이블의 deptno가 참조하고 있으므로 지워지지 않습니다.

이렇게 하는 이유?
1. 데이터의 품질을 높이기 위해
2. 조인할 때 outer join은 남발하지 않기 위해서 입니다.

Ex) emp와 dept테이블 생성스크립트를 수행합니다.

1. dept 테이블의 deptno에 primary key 제약을 겁니다.

alter table dept
add constraint dept_deptno_pk primary key(deptno);

2. emp테이블의 deptno에 foreign key 제약을 걸면서 dept테이블의 deptno를 참조하겠다.

alter table emp
add constraint emp_deptno_fk foreign key(deptno)
references dept(deptno); << 참조하겠다

insert into emp(empno, ename, sal, deptno)
values (1234, 'jack', 4500, 70);

명령의 8 행에서 시작하는 중 오류 발생 -
insert into emp(empno, ename, sal, deptno)
values (1234, 'jack', 4500, 70)
오류 보고 -
ORA-02291: 무결성 제약조건(SCOTT.EMP_DEPTNO_FK)이 위배되었습니

다- 부모 키가 없습니다

109 WITH절 사용하기 1(WITH ~ AS)

"복잡한 쿼리 내에 동일한 쿼리가 두번 이상 발생한 경우에 사용하면 좋은 성능을 보이는 SQL"

"테스트 테이블과 같이 임시로 사용하는 데이터를 가지고 테스트 할 때 유용한 SQL"

- **속도가 빠르다.**

Ex 1) 1부터 10까지의 숫자를 출력하는 SQL을 작성하시오.

```
select level as num1
      from dual
      connect by level <=10;
```

위의 결과를 테이블로 만들고 싶은데 테이블로 만드려면 dba에게 부탁을 해야하는데 안해준다면 with절로 내가 사용하는 SQL내에서만 임시로 테이블을 만들면 된다.

Ex 2) 위의 SQL을 with절로 변경하시오!

```
      with test_table as (select level as num1
                          from dual
                          connect by level <=10)

      select num1
            from test_table;
```

※ 설명 : as 다음에 나오는 괄호안의 쿼리문의 결과 데이터로 임시 테이블을 생성하는데 그 테이블 이름이 test_table이 됩니다.
그리고 그 아래의 쿼리에서 test_table 가지고 다양하게 쿼리문장을 만들어서 테스트 할 수 있습니다. 이 임시테이블은 with절이 끝나면 사라집니다.

- with 절의 장점? 똑같은 SQL을 하나의 SQL내에서 여러 개 사용될 때 유용합니다.

■ accept절 사용 방법

```
select empno, ename, sal
      from emp
      where empno = &사원번호;
```

입력 메시지 창에서 사원번호에 대한 값입력이라고 출력되는게 아니라 사원번호를 입력하세요~라고 메세지창의 메세지가 나오게 하고 싶다면?

```
accept p_num1 prompt '사원번호를 입력하세요~'
select empno, ename, sal
      from emp
      where empno = &p_num1;
```

110 WITH절 사용하기 2(SUBQUERY FACTORING)

" with절을 사용할 때 with 절 내에서 같은 SQL이 아래와 같이 두번이상 사용되게 되면 TEMP TABLE에 자동으로 저정되고 한번만 사용하면 with절의 as 다음에 나오는 쿼리문의 결과가 메모리에 저장된다."

```
explain plan for
with job_sumsal as (select job, sum(sal) as 토탈
                    from emp
                    group by job)


with job_sumsal as (select job, sum(sal) as 토탈
                    from emp
                    group by job)

select job, 토탈
      from job_sumsal
      where 토탈>(select avg(토탈)
                  from job_sumsal);
```

```
| Id  | Operation                          | Name                   | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                   |                        |    14 |   266 |     8  (13)| 00:00:01 |
|   1 |  TEMP TABLE TRANSFORMATION         |                        |       |       |            |          |
|   2 |   LOAD AS SELECT (CURSOR DURATION MEMORY)| SYS_TEMP_0FD9D666B_1FCBF8 |       |       |            |          |
|
|   3 |    HASH GROUP BY                   |                        |    14 |   266 |     4  (25)| 00:00:01 |
|   4 |     TABLE ACCESS FULL              | EMP                    |    14 |   266 |     3   (0)| 00:00:01 |
|*  5 |   VIEW                             |                        |    14 |   266 |     2   (0)| 00:00:01 |
|   6 |    TABLE ACCESS FULL               | SYS_TEMP_0FD9D666B_1FCBF8 |    14 |   266 |     2   (0)| 00:00:01 |
|   7 |    SORT AGGREGATE                  |                        |     1 |    13 |            |          |
|   8 |     VIEW                           |                        |    14 |   182 |     2   (0)| 00:00:01 |
|   9 |      TABLE ACCESS FULL             | SYS_TEMP_0FD9D666B_1FCBF8 |    14 |   266 |     2   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------------
```

```
select job, 토탈
      from job_sumsal;
```

116 SQL로 알고리즘 문제 풀기 6(사각형 출력)

Ex 1) ★을 아래와 같이 5개를 가로로 출력하시오.

★★★★★

```
select lpad('★', 5, '★')
      from dual;
```

Ex 2) 위의 별 5개가 아래로 아래와 같이 6개가 나오게 하시오

★★★★★
★★★★★
★★★★★
★★★★★
★★★★★
★★★★★

```
select lpad('★', 5, '★')
from dual
connect by level <=6;
```

121 SQL로 알고리즘 문제 풀기 11(최대 공약수)

"최대공약수가 필요한 이유? 약분을 빨리 하기 위해서 입니다."

16명의 인부들에게 24개의 빵을 공평하게 나눠주려면 1명당 몇개씩 주면 되는가?

```
    8 )16   24
    ------------      2명에게 3개의 빵을 나눠주는것과 같다.
       2    3         1.5개
```

select * from table(dbms_xplan.display);
이렇게 허면 임시 table을 쓰지 않는다.

```
-------------------------------------------------------------------------------
| Id  | Operation          | Name | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |    14 |   266 |     4  (25)| 00:00:01 |
|   1 |  HASH GROUP BY     |      |    14 |   266 |     4  (25)| 00:00:01 |
|   2 |   TABLE ACCESS FULL| EMP  |    14 |   266 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------
```

*with 절을 사용할 때 쓰는 유용한 힌트 2가지

1. /*+ materialize */ : temp table을 하드 디스크에 생성해라~
2. /*+ inline */ : temp table을 만들지 말고 그냥 from 절의 서브쿼리인 in line view를 써라

```
explain plan for
with job_sumsal as (select /*+ materialize */ job, sum(sal) as 토탈
                    from emp
                    group by job)
select job, 토탈
    from job_sumsal;

select * from table(dbms_xplan.display);
```

이렇게 하면 temp table이 생성된다. & 속도 빨라짐

```
explain plan for
with job_sumsal as (select /*+ inline */ job, sum(sal) as 토탈
                    from emp
                    group by job)
select job, 토탈
    from job_sumsal;
```

temp table을 만들기 싫으면 inline을 써준다.

11/13 (123+)

2020년 11월 13일 금요일 오전 9:54

- 기타 SQL기능들과 수업때 안다뤘던 주요 SQL기능들

복습 : 1. 기본 SQL문장
2. 함수
3. 조인문:
    1. 오라클 조인문법
        1. equi join
        2. non equi join
        3. outer join
        4. self join
    2. 1999 ANSI 문법 (하둡)
        1. using절을 사용한 조인
        2. natural join
        3. left/right/full outer 조인
        4. cross 조인
4. 집합연산자
5. 서브쿼리문
6. DML문
7. DDL문
8. TCL문
9. 계층형 질의문
10. WITH절

- 3개 이상의 테이블 조인
  연결고리 1 연결고리2
  dept----------emp----------salgrade

  where e.deptno = d.deptno and
  e.sal between s.losal and s.hisal

123 SQL로 알고리즘 문제 풀기 13(피타고라스의 정리)

문제 풀기:

함수 :

- **단일행함수 : 문자, 숫자, 날짜, 변환, 일반**
  숫자함수 : 1. round
  2. trunc
  3. mod
  4. power
  2^3 구하는법

```
select power(2,3)
    from dual
```

- **복수행함수**

- **외부테이블 가 있는지 확인하고 삭제하는 방법**

```
select table_name, external
    from user_tables
    where external = 'YES';

    drop table EXT_EMP11;
    drop table EXT_DEPT;
```

- 외부 테이블 (external table)
- 오라클 테이블의 종류?

1. 일반 테이블(heap table)
2. 임시 테이블(temp table)
3. 외부 테이블(external table)
4. 파티션 테이블(partition table)

면접질문 ? 병렬처리로 프로그래밍 해본적이 있나요?
답변 : SQL로는 파티션 테이블 만들어서 병렬 쿼리를 이용해서 병렬처리를 해보았습니다.

외부테이블(External table)?
Database 외부에 있는 엑셀 파일이나 csv파일, text 파일을 insert문으로 만들어서 데이터베이스에 입력하지 않고도 그냥 링크만 걸어서 해당 파일의 데이터를 select 할 수 있는 테이블

ex) 1. emp.txt를 c드라이브 밑에 data라는 폴더를 만들고 그 안에 emp.txt를 넣으시오!

2. c드라이브에 data폴더를 오라클 디렉토리로 생성하시오.
   create directory emp_dir as 'c:₩data';

3. 외부테이블을 생성하시오.

```
create table ext_emp11
    ( EMPNO    NUMBER(10),
      ENAME    VARCHAR2(20),
      JOB      VARCHAR2(20),
      MGR      NUMBER(10),
      HIREDATE  DATE,
      SAL    NUMBER(10),
      COMM    NUMBER(10),
      DEPTNO  NUMBER(10))
      organization external <---external table 생성하겠다
     (type oracle_loader <--data를 로드하는 엔진을 sql*developer를 사용
      default directory emp_dir <---emp.dlr를 기본 디렉토리로 하겠다.
      access parameters <---text 파일의 내용을 입력하기 위한 문법을
                           실행하겠다.
      (records delimited by newline <--- 행과 행사이는 엔터로 구분하겠다.
```

fields terminated by "," <--- 데이터와 데이터는 콤마(,)로 구분하겠다.
(empno char, <--외부테이블의 컬럼들의 실제 데이터는 문자
ename char, <--but 오라클에는 숫자로 입력한다
job char,
mgr char,
hiredate date "yyyy/mm/dd",
sal char,
comm char ,
deptno char))
location ('emp.txt')); <---링크할 텍스트 파일 이름

4. select * from ext_emp11; 해본다.

대용량 엑셀 파일이나 text파일을 db에 넣으려면 시간이 많이 걸립니다.

- 외부테이블의 장점?

1. database에 저장공간이 필요없다.
2. 대용량 데이터인 경우 테이블에 입력하는 시간을 줄일 수 있다.

- 외부테이블의 단점?

1. DML작업이 안되고 오직 SELECT만 된다.
2. 인덱스를 생성할 수 없습니다.

■ 정규식 함수 5가지

"데이터 검색을 좀더 상세하게 하려 할때 기존에 함수로는 표현할 수 없는
검색을 할 수 있게 해주는 함수

regular expression(정규 표현식)

정규 표현식 코드는 오라클 뿐만 아니라 다른 언어에서도 공통적으로 사용하는 표현식 코드입니다.
오라클 10.1버전부터 정규 표현식을 지원합니다.
오라클 데이터베이스는 정규표현식의 POSIX 연산자를 | 원합니다. POSIX는 Portable Operation System Interface의 약자로 시스템간 호환성을 위해 미리 정의된 인터페이스를 의미한다.

POSIX연산자에는 기본연산자, 앵커 수량사, 서브표현식, 역참조, 문자리스트, posix 문자 클래스등이 있습니다.

1. 기본연산자

연산자	영문	설명
.	dot	모든 문자와 일치
|	or	대체 문자를 구분
₩	backslash	다음 문자를 일반 문자로 처리

```
Ex ) select regexp_substr('aab', 'a.b') as c1,
            regexp_substr('abb', 'a.b') as c2,
            regexp_substr('acb', 'a.b') as c3,
            regexp_substr('adc', 'a.b') as c4
            from dual;
```

regexp_substr('aab', 'a.b') as c1
a.b - 앞에가 a고 뒤에가 b인데 .에 아무거나 와도 상관 없다
--> aab에서 'a.b'부분을 잘라내라

```
C1     C2     C3     C4
------ ------ ------ -----
aab    abb    acb
```

- 서브표현식

서브표현식은 표현식을 소괄호로 묶은 표현식입니다.
서브표현식은 하나의 단위로 처리됩니다.

연산자	설명
(표현식)	괄호 안의 표현식을 하나의 단위로 취급
+	1회 또는 그 이상의 횟수로 일치

```
Ex ) select regexp_substr('ababc', '(ab)+c') as c1,
            regexp_substr('ababc', 'ab+c') as c2,
            regexp_substr('abd', 'a(b|c)d') as c3,
            regexp_substr('abd', 'ab|cd') as c4
            from dual;
```

- regexp_count 함수

특정 단어나 철자가 문장에서 반복되엇 출력되는지 확인하는 함수
count ----------> regexp_coun
업그레이드
테이블의 행의 건수를 세는 함수 겨울왕국에서 elas가 몇번 나오는지 알 수 있다.(쉽게)

아래의 긴 문장에서 gtc라는 단어가 몇번 나오는가?

```
SELECT REGEXP_COUNT(
'ccacctttccctccactcctcacgttctcacctgtaaagcgtcctccctcatccccatgccccttaccctgcag
ggtagagtaggctagaaaccagagagctccaagctccatctgtggagaggtgccatccttgggctgcagag
agaggagaatttgccccaaagctgcctgcagagcttcaccacccttagtctcacaaagccttgagttcatagca
tttcttgagttttcaccctgcccagcaggacactgcagcacccaaagggcttcccaggagtagggttgccctca
agaggctcttgggtctgatggccacatcctggaattgttttcaagttgatggtcacagccctgaggcatgtagg
ggcgtggggatgcgctctgctctgctctcctctcctgaaacccctgaaccctctggctaccccagagcacttaga
gccag',
'gtc') AS Count
FROM dual;
```

4번

- regexp_replace
  replace ----------> regexp_replace

  Ex) 이름과 월급을 출력하는데 월급을 출력할 때 replace 함수를 이용해서 숫자0을 *로 출력하시오.

```
select ename, replace(sal, 0, '*')
    from emp;
```

  Ex) 이름과 월급을 출력하는데 월급 숫자0~3까지를 *로 출력하시오!
  ([0-3]: 0~3까지를 의미합니다.)

```
select ename, regexp_replace(sal, '[0-3]', '*')
    from emp;
```

```
KING        5***
BLAKE       *85*
CLARK       *45*
JONES       *975
MARTIN      **5*
ALLEN       *6**
TURNER      *5**
JAMES        95*
WARD         **5*
FORD         ****
SMITH         8**
SCOTT         ****
ADAMS          ****
MILLER         ****
```

- 다중 insert문

```
insert into emp(empno, ename, sal)
    values ('1234', 'SCOTT', 3000);
```

  그동안 배웠던 insert문장은 한번에 한건만 입력할 수 있었습니다.
  다중 insert문을 이용하면 여러개의 테이블에 동시에 같은 데이터를 여러 개 입력할 수 있습니다.

- 다중 insert 문의 종류 4가지

1. 무조건 all insert문
2. 조건부 all insert문
3. 조건부 first insert문
4. pivoting insert문

1. 무조건 all insert 문

"여러개의 테이블에 조건 없이 한번에 데이터를 입력하는 것"

```
create table targer_a
    as
    select *
        from emp
        where 1=2;

create table targer_b
    as
    select *
        from emp
        where 1=2;
create table targer_c
    as
    select *
        from emp
        where 1=2;
```

설명 : where절에 1=2가 거짓이므로 emp테이블을 가져와서 target-a(b/c) 테이블을 생성할 때 데이터는 못 가져오고 구조만 가져와서 만든다.

```
insert all into target_a
        into target_b
        into target_c
select *
    from emp;
```

2. 조건부 all insert문

조건에 맞는 데이터만 입력되게 조건을 주는 입력문

Ex) target_a 테이블에는 comm을 받는 사원들만 입력하고
target_b 테이블에는 comm을 받지 않는 사원들만 입력하시오.

```
Insert all
    when comm is not null then
    into target_a (empno, ename, sal, comm)
    when comm is null then
    into target_b(empno, ename, sal, comm)
select empno, ename, sal, comm
    from emp;
```

3. 조건부 first insert문

조건에 맞는 데이터가 첫번째 테이블에 입력되고 남은 나머지 데이터를 가지고 새로운 조건에 맞춰서 두번째 또는 세번째에 입력하는 insert문

Ex) 부서번호가 20번인 사원들은 target_a에 입력하고 남은 나머지 부서번호인 사원들의 월급중에서 월급이 1200 이상은 target_b에 입력하고 1200보다 작은 사원은 target_c에 입력하시오!

```
truncate table target_a;
truncate table target_b;
truncate table target_c;

insert first
   when deptno = 20 then into target_a
   when sal>=1200 then into target_b
   when sal < 1200 then into target_c
select *
   from emp;
```

124 SQL로 알고리즘 문제 풀기 14(몬테카를로 알고리즘)

몬테카를로 알고리즘은 수많은 노가다를 통해서 정답을 알아내는 것을 말한다.

SQL 게시판에 게시글 1128
몬테카를로 알고리즘으로 원주율을 알아내세요~ 보기

Ex) 삼성생명 - 몬테카를로 알고리즘을 이용해서 데이터 분석을 했는데 금융사고가 생겼을 때를 대비해서 최대 얼마만큼의 자금을 보유하고 있어야 하는가?

11/16

2020년 11월 16일 월요일 오전 9:59

4. **pivoting insert문**
   "어렵게 pivot문으로 작성하지 않고 pivoting insert문으로 데이터를 입력한 후에 pivot문을 사용하지 않은 간단한 SQL로 결과를 보게할 때 필요한 insert문"

   Ex) create table week_sum_sal
   (empno number(10),
   ename varchar2(10),
   mon number(10),
   tues number(10),
   wed number(10),
   thur number(10),
   fri number(10));

   insert into week_sum_sal
   select empno, ename, sal*0.1, sal*0.2, sal*0.3, sal*0.4, sal*0.5
   from emp;

- SQL 복습

1. 기본 select 문장
2. 함수: 단일행 함수, 복수행 함수, 데이터 분석 함수
3. 조인
4. 집합 연산자
5. 서브쿼리문
6. DML문장
7. DDL문장
8. TCL문장
9. 계층형 질의문
10. with절
11. 제약
12. database object 5가지
13. flashback 기능
14. 다중 insert문 4가지

■ DCL 문장

SQL문장의 종류? 1. 쿼리문 : select 절의 6가지 절
2. DML문: insert, update, delete, merge
3. DDL문 : create, alter, drop, truncate, rename
4. DCL문 : grant, revoke
5. TCL문 : commit, rollback, savepoint

- 유저생성하는 방법
  create user king <--- user이름
  identified by tiger; <--- 패스워드

- 접속할 수 있는 권한을 부여하는 방법
  grant connect to king;

  grant dba to king; ---> 이렇게 하면 king에게 최고 권한을 준 것이다. /
  다른 권한을 주지 않아도 table생성 가능

  도스창 열고 king으로 접속을 해본다.
  sqlplus king/tiger
  show user --> king

- 내가 가지고 있는 시스템 권한 확인하는 방법
  select*
  from session_privs;
  ---
  allen이 가지고 있는 권한:
  SET CONTAINER
  CREATE TABLE <--- 테이블 생성 권한
  CREATE SESSION <--- 접속할 수 있는 권한

- 오라클에 존재하는 권한 리스트를 확인하는 방법

```
SQL> exit
C:₩Users₩stu03>sqlplus "/ as sysdba"
SQL> select *
        from session_privs;
```

※ 설명 : database에 있는 모든 테이블을 다 볼 수 있는 권한 select any table 권한 입니다.

- 권한 취소하기(revoke)
  sys 유져에서 allen에게 주었던 create table 권한을 취소하려면?
  SQL> revoke create table from allen;

- 유져 삭제하는 방법
  sys유져에서 아래와 같이 작업하면 됩니다.

  SQL> drop user allen cascade;

- database에 있 유져목록 확인하는 방법
  SQL > select username
  from dba_users;

오늘의 문제 (1-250)

2020년 10월 21일 수요일 오전 10:13

<오늘의 문제>

1. **이름과 월급과 직업을 출력하시오**
   select ename, sal, job from emp;

2. **이름과 입사일과 부서번호를 출력하시오**
   select ename, hiredate, deptno from emp;

3. **Emp 테이블의 컬럼이 무엇무엇이 있는지 확인하시오.**
   describe emp

4. **이름, 월급, 커미션을 출력하시오**
   select ename, sal, comm
   from emp; (from emp는 한줄에 같이 써도 되지만 다음 줄에 쓰는 것이 좋음)

5. **이름, 직업, 입사일, 부서번호를 출력하시오.**
   select ename, job, hiredate, deptno
   from emp;

6. **emp 테이블의 모든 컬럼을 검색하시오**
   Select empno, ename, sal, job, hiredate, mgr, deptno, comm
   From emp;

7. **Dept 테이블의 모든 컬럼을 출력하시오.**
   Select *
   From dept;

❖ 설명 : dept는 부서테이블이고 컬럼은 3개가 있는데 다음과 같습니다.
deptno: 부서 번호
dname: 부서명
loc : 부서 위치

8. **이름과 입사일과 부서번호를 출력하는데 컬럼명이 한글로 이름, 입사일, 부서번호로 출력되게 하시오.**
   Select ename as 이름, hiredate as 입사일, deptno as 부서번호
   From emp;

   Or

   Select ename 이름, hiredate 입사일, deptno 부서번호

From emp;

9. **이름과 월급을 출력하는데 컬럼명이 아래와 같이 출력되게 하시오.**
   Employee name, Salary

   답 : select ename as "Employee name", sal as "salary"
   From emp;

❖ 설명 : 컬럼 별칭에 대소문자를 구분하고 싶다거나 공백 문자나 특수문자를 넣으려면 양쪽에 "" (double quotation mark)를 둘러줘야 합니다.

10. **아래와 같이 결과를 출력하시오!**

    예시 : King의 직업은 president 입니다.
    Scott의 직업은 analyst 입니다.

    select name ||'의 직업은' ||job||'입니다.'
    From emp;

11. **위의 결과에서 출력되는 컬럼명을 사원정보라는 한글로 출력되게 하시오.**
    select name ||'의 직업은' ||job||'입니다.' as 사원정보
    From emp;

12. **부서번호를 출력하는데 중복을 제거해서 출력하시오.**
    Select distinct deptno
    From emp;

13. **이름과 월급을 출력하는데 월급이 높은 사원부터 출력하시오!**
    Select ename, sal
    From emp
    Order by sal desc;

14. **이름과 입사일을 출력하는데 최근에 입사한 사원부터 출력하시오.**
    Select ename, hiredate
    From emp
    Order by hiredate desc;

15. **이름과 직업과 입사일을 출력하는데 직업은 ABCD순으로 정렬해서 출력하고**
    직업을 ABCD순으로 정렬한 것을 기준으로 입사일을 먼저 입사한 사원부터 출력되게 하시오.
    select ename, job, hiredate
    from emp
    order by 2 asc, 3 asc;

16. **사원 번호가 7788번인 사원의 사원번호와 이름과 월급을 출력하시오**
    select empno, ename, sal

```
from emp
where empno = 7788;
```

17. **30번 부서번호에서 근무하는 사원들의 이름과 월급과 부서번호를 출력하시오!**

```
select ename, sal, deptno
from emp
where deptno = 30;
```

18. **위의 결과를 다시 출력하는데 월급이 높은사원부터 출력하시오!**

```
select ename, sal, deptno
from emp
where deptno = 30
Order by sal desc;
```

19. **부서번호가 20번인 사원들의 이름과 입사일과 부서번호를 출력하는데 최근에 입사한 사원부터 출력하시오!**

```
select ename, hiredate, deptno
from emp
where deptno = 20
order by hiredate desc;
```

20. **직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하시오!**

```
select ename, sal, job
from emp
where job = 'SALESMAN';
```

21. **81년 11월 17일에 입사한 사원의 이름과 입사일을 출력하시오.**

```
select ename, hiredate
from emp
where hiredate = '81/11/17';
```

❖ 설명 : 날짜 검색을 할 때는 년도/월/일 순으로 검색을 하면 되는데 나라마다 순서가 다를 수 있다.

미국이나 영국에서의 날짜 검색은 일/월/년도 순서 입니다.

22. **이름과 연봉(sal*12) 을 출력하는데 컬럼명을 한글로 이름, 연봉으로 출력하시오**

```
select ename as 이름, sal*12 as 연봉
from emp;
```

23. **위의 결과를 다시 출력하는데 연봉이 36000인 사원들의 이름과 연봉을 출력하시오.**

```
Select ename as 이름, sal*12 as 연봉
From emp
Where 연봉 = 36000;
```

------> 실행 안됨

❖ 설명 : 위의 SQL이 수행되지 않는 이유는 실행 순서 때문입니다.

코딩순서 : select ----> from ----> where
오라클 실행순서 : from ----> where ----> select
그래서 실행되지 않는 것임

하지만 order by 는 코딩 순서, 오라클 실행순서 모두 맨 마지막이라서 order by 다음에 연봉이라고 넣어도 답이 나옴.

맞는 답 :

```
select ename as 이름, sal*12 as 연봉
from emp
where sal*12 = '36000';
```

24. 이름과 연봉을 출력하는데 연봉이 높은 사원부터 출력하시오! (위의 문제와 연관)

```
Select ename as 이름, sal*12 as 연봉
From emp
Order by 연봉 desc;
```

이렇게도 가능

❖ 코딩순서 : select ----> from ----> order by
실행순서 : from ----> select ----> order by

25. 아래의 SQL에서 더하기가 먼저 실행되게 하시오.

<보기>

```
Select ename, sal + 200*2
From emp;
```

답변 : 괄호를 사용해야 합니다.

```
Select ename, (sal + 200)*2
From emp;
```

26. 커미션이 150 이상인 사원들의 이름과 커미션을 출력하시오!

```
select ename, comm
  2  from emp
  3  where comm >= 150;
```

27. 월급이 3000이상인 사원들의 이름과 월급을 출력하시오.

```
select ename, sal
from emp
where sal >= '3000';
```

28. 직업이 SALESMAN이 아닌 사원들의 이름과 직업을 출력하시오!

```
select ename, job
from emp
where job != 'SALESMAN';
```

- ❖ 문자와 날짜는 반드시 양쪽에 싱글 쿼테이션 마크를 둘러줘야 한다.
- ❖ 더블쿼테이션 마크는 오라클 전체를 통틀어서 딱 하나의 케이스에서만 사용하는데 컬럼별칭 사용할 때 공백문자, 특수문자, 대소문자를 구분해서 컬럼별칭을 출력하고자 할 때

29. 월급이 2400이하인 사원들의 이름과 월급을 출력하는데 월급이 높은 사원부터 출력하시오!

```
select ename, sal
from emp
where sal <= 2400
order by sal desc;
```

30. 월급이 1000에서 3000사이인 사원들의 이름과 월급을 출력하시오!

```
Select ename, sal
From emp
Where sal between 1000 and 3000;
```

또는

```
Select ename, sal
From emp
Where sal >=1000 and sal <=3000;
```

31. 월급이 1000에서 3000사이가 아닌 사람들의 이름과 월급을 출력하시오!

```
select ename, sal
from emp
where sal not between 1000 and 3000;
```

32. 1981년도에 입사한 사원들의 이름과 입사일을 출력하시오.

```
select ename, hiredate
from emp
where hiredate between '81/01/01' and '81/12/31';
```

33. 이름의 끝 글자가 T로 끝나는 사원들의 이름을 출력하시오!

```
select ename
from emp
where ename like '%T';
```

34. 81년도에 입사한 사원들의 이름과 입사일을 출력하는데 between and를 사용하지 말고 like를 사용하시오.

```
select ename, hiredate
from emp
where hiredate like '81%';
```

35. 이름의 두번째 철자가 M인 사원들의 이름을 출력하시오!

```
Select ename
From emp
Where ename like '_M%'
```

❖ 설명 :

Like와 같이 쓸 수 있는 키워드가 2개가 있다.

%: 와일드 카드 - 이 자리에 뭐가 와도 관계없고 그 갯수가 몇 개가 돼도 관계 없다.

_ : 언더바 - 이 자리에 뭐가 와도 관계 없는데 자릿수는 한개여야 된다.

36. 이름의 세번째 철자가 A인 사원들의 이름을 출력하시오!

```
select ename
from emp
where ename like '__A%';
```

37. 이름의 첫번째 철자가 A로 시작하는 사원들의 이름과 월급을 출력하시오. (10/22)

```
select ename, sal
from emp
where ename like 'A%';      (where + 검색조건)
```

❖ 설명 :

%가 특수문자 %가 아니라 와일드 카드로 인식하려면 기타 비교연산자중에 하나인 like를 사용해야 합니다.

❖ like 연산자의 키워드 2가지?

1. % : 와일드 카드 : 이 자리에 뭐가 와도 관계없고 그 갯수가 몇 개가 되든 관계 없다.
2. _ : 언더바 : 이 자리에 뭐가 와도 관계 없는데 자릿수는 한개여야 한다.

38. 이름이 SCOTT인 사원의 이름과 월급과 직업을 출력하시오!

```
select ename, sal, job
from emp
where ename = 'SCOTT';      (이름이 정확히 SCOTT이니까 이 경우에는 '=' 사용가능)
```

39. 이름의 끝에서 두번째 철자가 T인 사원들의 이름과 월급을 출력하시오!

```
select ename, sal
from emp
where ename like '%T_';
```

40. 커미션이 null인 사원들의 이름과 커미션을 출력하시오.

흔히하는 오답:

```
select ename, comm
```

```
from emp
where comm = null
```

(정답):

```
select ename, comm
from emp
where comm is null
```

❖ 설명 :

comm=null로는 조회할 수 없다. 왜냐하면 null은 알 수 없는 값이므로 비교 연산자인 = 로는 조회할 수없다. 기타 비교 연산자인 is null을 사용하여야 한다.

41. 커미션이 null이 아닌 사원들의 이름과 커미션을 출력하시오.

```
select ename, comm
from emp
where comm is not null;
```

42. mgr(관리자의 사원번호)이 null인 사원의 이름과 직업을 출력하시오.

```
select ename, job
from emp
where mgr is null;
```

설명 : 사장(president)은 관리자 번호(mgr)가 없다.

43. 사원번호, 이름, 관리자 번호(mgr)을 출력하시오!

```
select empno, ename, mgr
from emp;
```

44. 직업이 SALESMAN, ANALYST인 사원들의 이름과 직업을 출력하시오.

```
select ename, job
from emp
where job in ('SALESMAN', 'ANALYST');
```

꼭 괄호로 묶어주고, 문자니까 싱글 쿼테이션 넣어줘야 함.

45. 직업이 SALESMAN, ANALYST가 아닌 사원들의 이름과 직업을 출력하시오.

```
select ename, job
from emp
where job not in ('SALESMAN', 'ANALYST');
```

46. 직업이 SALESMAN이 아닌 사원들의 이름과 월급과 직업을 출력하시오!

```
select ename, sal, job
from emp
where job != 'SALESMAN';
```

47. 위의 결과를 다시 출력하는데 월급이 높은 사원부터 출력하시오.

```
select ename, sal, job
from emp
where job != 'SALESMAN'
order by sal desc;
```

48. 우리반 테이블에서 이름과 나이를 출력하는데 나이가 높은 학생부터 출력하시오.

```
select ename, age
from emp12
order by age desc;
```

49. 이름과 나이와 주소를 출력하는데 30살 이상인 학생들만 출력하시오.

```
select ename, age, address
from emp12
where age >=30;
```

50. 성씨가 김씨인 학생들의 이름과 통신사를 출력하시오.

```
select ename, telecom
from emp12
where ename like '김%';
```

51. 전공에 통계를 포함하고 있는 학생들의 이름과 전공을 출력하시오!

```
select ename, major
from emp12
where major like '%통계%';
```

❖ 설명: like 연산자를 사용할 때 특정 단어를 포함하고 있는 데이터를 검색하려면 %단어% 라고 하면 됩니다.

52. 우리반에 gmail을 사용하는 학생들의 이름과 메일을 출력하시오.

```
select ename, email
from emp12
where email like '%gmail%';
```

53. 나이가 27에서 34 사이인 학생들의 이름과 나이를 출력하시오.

```
select ename, age
from emp12
where age between 27 and 34;
```

54. 나이가 27에서 34 사이가 아닌 학생들의 이름과 나이를 출력하시오.

```
select ename, age
from emp12
where age not between 27 and 34;
```

55. 주소가 경기도인 학생들의 이름과 나이와 주소를 출력하시오.

```
select ename, age, address
from emp12
```

where address like '경기도%';

56. 통신사가 sk, lg인 학생들의 이름과 통신사를 출력하시오.

```
select ename, telecom
from emp12
where telecom in ('sk','lg');
```

57. 서울에서 사는 학생들의 이름과 나이와 전공을 출력하는데 나이가 높은 학생부터 출력하시오!

```
select ename, age, major
from emp12
where address like '%서울%'
order by age desc;
```

58. 이메일이 gmail이 아닌 학생들의 이름과 이메일을 출력하시오.

```
select ename, email
from emp12
where email not like '%gmail%'
```

59. 아래와 같이 결과가 출력되게 하시오.

김주원 학생의 나이는 44세 입니다.
권세원 학생의 나이는 36세 입니다.
.
.
.

```
select ename||'학생의 나이는 '||age||'입니다'
from emp12
order by age desc;
```

60. 직업이 SALESMAN 이거나 또는 ANALYST인 사원들의 이름과 월급과 직업을 출력하시오.

```
select ename, sal, job
from emp
where job = 'SALESMAN' or job = 'ANALYST';
```

또는

```
select ename, sal, job
from emp
where job in ('SALESMAN', 'ANALYST');
```

61. 성씨가 김씨, 이씨가 아닌 학생들의 이름을 출력하시오!

```
select ename
from emp12
```

```
where ename not like '김%' and ename not like '이%';
```

62. 이메일이 gmail과 naver가 아닌 학생들의 이름과 이메일을 출력하시오.

```
select ename, email
from emp12
where email not like '%gmail%' and email not like '%naver%';
```

63. 우리반 테이블에서 통신사가 sk와 관련된 통신사이면 그 학생의 이름과 통신사를 출력하시오! 정확하게 데이터가 출력이 되어지게끔 SQL을 작성하세요.

```
select ename, upper(telecom)
from emp12
where upper(telecom) like ('%SK%');
```

Δ

telecom 데이터를 전부 대문자로 변경해버린다.

sK ----> SK
Sk ----> SK
sk ----> SK

또는

```
select ename, telecom
from emp12
where upper(telecom) like ('%SK%');
```

이것도 가능

64. 성씨가 김, 이, 유씨 인 학생들의 이름과 나이를 출력하는데 like 쓰지말고 in 과 substr을 써서 출력하시오!

```
select ename, age
from emp12
where substr(ename, 1, 1) in ('유', '김', '이');
```

65. 우리반 테이블에서 이메일과 이메일 철자의 길이를 출력하는데 이메일 철자의 길이가 가장 긴것부터 출력하시오!

```
select ename, email, length(email)
from emp12
order by length(email) desc;
```

66. emp 테이블에서 ename을 출력하고 그 옆에 ename의 첫번째 철자를 출력하시오!

```
select ename, substr(ename, 1, 1)
from emp;
```

67. 위의 결과를 다시 출력하는데 이름의 첫번째 철자로 출력되는 부분을 소문자로 출력하시오!

```
select ename, lower(substr(ename, 1, 1))
from emp;
```

또는

```
select ename, substr(lower(ename),1,1)
from emp;
```

68. 아래의 결과를 initcap 쓰지 말고, upper, lower, substr, || 를 사용해서 출력하시오!

```
select initcap(ename)
from emp;
```

답:

```
select substr(Upper(ename),1,1)||substr(lower(ename),2,10)
from emp;
```

설명 : substr(첫번째 인자값, 두번째 인자값, 세번째 인자값,)

substr(ename, 2 , 2)

Δ	Δ
잘라낼 시작 자리수	잘라낼 자리수 개수

substr 작성시 세번째 인자값에 아무것도 적지 않으면 두번째 인자값의 자리부터 뒤에 값이 다 나옴.

예제 : select substr('smith', 1, 3)
from dual;

dual : select 절에 있는 함수의 값을 보기위한 가상의 테이블
emp쓰면 테이블의 갯수만큼 나온다. (14개)

```
 s  m  i  t  h
 1  2  3  4  5
-5 -4 -3 -2 -1
```

끝에있는 철자를 잘라내고 싶을 때는 두번째 인자값에 마이너스 값을 넣는 것이 좋음

69. 사원 테이블에서 이름을 출력하고 그 옆에 이름의 끝철자를 출력하는데 끝 철자를 소문자로 출력하시오. (10/23)

```
select ename, lower(substr(ename, -1, 1))
from emp;
```

70. 이메일을 출력하고 그 옆에 이메일의 도메인을 출력하시오.

도메인 : gmail, naver

```
select email, substr(email, instr(email, '@')+1, instr(substr(email,instr(email,'@')), '.')-2)
from emp12;
```

또는

```
select email, substr(email,instr(email,'@')+1,length(email)-instr(email,'@')-4)
from emp12;
```

또는

```
select ename, substr(email, instr(email, '@') +1, instr(email, '.', -1, 1) - instr(email, '@') -1)
  from emp12;
```

Δ 뒤에서부터 센다.

71. 사원 테이블에서 이름과 월급을 출력하는데

숫자 0을 *로 출력하시오.

```
select ename, replace(sal, '0', '*')
from emp;
```

72. 우리반 테이블에서 이름을 출력하고 그 옆에 이름에 두번째 철자를 출력하시오!

```
select ename, substr(ename, 2,1)
from emp12;
```

73. 아산병원의 전광판을 구현하시오!

유*수

송*미

```
select ename, replace(ename, substr(ename, 2,1), '*')
from emp12;
```

74. 남궁솔미 데이터를 입력하고 남궁*미로 출력 되게 위의 SQL을 다시 작성하시오!

유*수

송*미

남궁*미

```
select ename, replace(ename, substr(ename, -2,1), '*')
from emp12;
```

75. 경기도에 사는 학생의 이름과 주소를 출력하시오! 와일드카드를 양쪽에 사용하지 말고 한쪽에만 사용해서 출력하시오!

(SQL 튜닝때 배울건데 와일드 카드를 양쪽에 사용해서 데이터 검색을 하면 검색속도가 느려집니다.)

```
select ename, address
from emp12
where ltrim(address) like '경기도%';
```

76. 정보통계학과가 전공인 학생의 이름과 나이와 전공을 출력하시오!

```
select ename, age, major
from emp12
where ltrim(major) like '정보통계학과%';
```

77. 우리반 나이의 평균값을 출력하시오.

```
select avg(age)
  from emp12;
```

78. 위의 결과를 반올림해서 소수점 이후는 안 나오게 하시오.

```
select round (avg(age), 0)
    from emp12;
```

79. 우리반에서 나이가 짝수인 학생들의 이름과 나이를 출력하시오.

```
select ename, age
    from emp12
    where mod(age,2)=0;
```

80. select ename, sysdate - hiredate
from emp;
위의 결과에서 소수점 이하는 안나오게 반올림 하시오.

```
select ename, round(sysdate - hiredate,0)
from emp;
```

81. 이름, 입사한 날짜부터 오늘까지 총 몇주 근무했는지 출력하시오!
(소수점 이하 안나오게 반올림도 하시오)

```
select ename, round((sysdate - hiredate)/7,0)
from emp;
```

또는

```
select ename, round(round(sysdate - hiredate)/7)
from emp;
```

82. 이름, 입사한 날짜부터 오늘까지 총 몇 달 근무했는지 출력하시오

```
select ename,
       months_between(sysdate, hiredate)
from emp;
```

❖ 설명 : months_between(최신날짜, 옛날날짜)

날짜와 날짜 사이의 개월수를 출력

83. 아래와 같이 결과를 출력하시오!

KING은 467달을 근무했습니다.

Blake은 468달을 근무했습니다.

.

.

.

```
select ename||'은 '||round(months_between(sysdate,hiredate),0)||'달을 근무했습니다.'
from emp;
```

84. 오늘 날짜에서 100달 뒤에 돌아오는 돌아오는 목요일 날짜를 출력하시오!

```
select next_day(add_months(sysdate, 100), '목요일')
from dual;
```

85. 81/11/17에 입사한 사원의 이름과 입사일을 출력하시오.

```
select ename, hiredate
    from emp
    where hiredate = '81/11/17';
```

86. 이름, 입사일, 입사한 요일을 출력하시오.

```
select ename, hiredate, to_char(hiredate, 'day')
   from emp;
```

87. 수요일에 입사한 사원들의 이름과 입사일을 출력하시오!

```
select ename, to_char(hiredate, 'day')
    from emp
    where to_char(hiredate, 'day') = '수요일';
```

88. 내가 무슨 요일에 태어났는지 확인하시오!

```
select to_char(to_date('94/06/30', 'RR/MM/DD'), 'DAY')
from dual;
```

❖ 설명 : to_date 함수로 날짜로 변환해줘야 합니다.

89. 이름, 입사한 요일을 출력하는데 입사한 요일이 월화수목금토 순으로 정렬되어서 출력되게 하시오!

```
select ename,to_char(hiredate, 'day')
    from emp
    order by to_char(hiredate-1, 'd') asc;
```

또는

```
select ename, to_char(hiredate, 'day')
```

```
from emp
order by replace(to_char(hiredate,'d'), 1, 8) asc;
```

90. 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하는데 월급이 높은 사원부터 출력하시오! (10/26)

```
select ename, sal, job
    from emp
    where job = 'SALESMAN'
    order by sal desc;
```

❖ 설명: order by 절은 항상 맨 마지막에 실행이 된다.
select문장에서도 맨 마지막에 작성한다.

91. 직업이 SALESMAN이 아닌 사원들의 이름과 입사일과 직업을 출력하는데 최근에 입사한 사원부터 출력하시오.

```
select ename, hiredate, job
    from emp
    where job != 'SALESMAN' (검색조건)  !=, <> 다 써도됨
    order by hiredate desc;
```

❖ 설명 : 3 select 컬럼명
1 from table명
2 where 검색조건
4 order by 정렬할 컬럼명
Δ
실행순서

92. 월급이 1000에서 3000 사이인 사원들의 이름과 월급을 출력하시오.

```
select ename, sal
    from emp
    where sal between 1000 and 3000;
```

93. 이름을 출력하고 그 옆에 이름의 첫번째 철자만 출력하는데 소문자로 출력하시오!

```
select ename, lower(substr(ename, 1,1))
        from emp;
```

94. 우리반 테이블에서 이름과 이메일을 출력하고 그 옆에 이메일에서 @가 몇번째 철자인지 출력하시오.

```
select ename, email, instr(email, '@')
    from emp12;
```

95. 이름과 입사일, 입사한 년도를 4자리로 출력하시오.

```
select ename, hiredate, to_char(hiredate, 'RRRR')
```

```
from emp;
```

❖ 설명 : 년도 : RRRR, RR, YYYY, YY
달 : MM, MON
일 : DD
시간 : HH, HH24
분 : MI
초 : SS
요일 : DAY, DY, D

```
select ename, hiredate, to_char(hiredate, 'MM')
   from emp;
```
---> '달' 만 추출

to_char : 날짜 ---> 문자, 숫자 ---> 문자

96. 11월에 입사한 사원들의 이름과 입사일을 출력하시오!

```
select ename, hiredate
    from emp
    where to_char(hiredate, 'Mon') = '11월';
```

or

```
select ename, hiredate
   from emp
   where to_char(hiredate, 'mm') = '11';
```

('11' 처럼 숫자 앞뒤로 싱글 쿼테이션을 붙이는게 좋다.)

97. 문제 96번을 to_char 이용하지 말고 substr로 수행하시오.

```
select ename, hiredate
    from emp
    where substr(hiredate, 4, 2) = '11';
```

98. 1981년도에 입사한 사원들의 이름과 입사일을 출력하시오.

```
select ename, hiredate
    from emp
    where hiredate between to_date('81/01/01', 'RR/MM/DD') and to_date('81/12/31',
'RR/MM/DD');
```

또는

```
select ename, hiredate
    from emp
    where to_char(hiredate,'RRRR') = '1981';
```

99. 1981년도에 입사한 사원들의 이름과 입사일과 입사한 년도를 출력하는데, 가장 최근에 입사한 사원부터 출력하시오.

```
select ename, hiredate, to_char(hiredate, 'RRRR')
    from emp
    where to_char(hiredate, 'RRRR') = '1981'
    order by hiredate desc
```

100. 이름, 커미션을 출력하시오.

```
select ename, comm
    from emp;
```

101. 이름, 커미션을 출력하는데 커미션이 null인 사원들은 no comm이라는 글씨로 출력되게 하시오.

```
select ename, nvl(to_char(comm),'no comm')
    from emp;
```

❖ 설명 : 숫자형을 문자형으로 변환해서 데이터 타입을 서로 동일하게 맞춰주고 출력하면 된다.

nvl 다음에 오는 것들이 (문자형, 문자형)이거나 (숫자형, 숫자형) 이런식으로 바뀌어 줘야함.
따라서, 앞에오는 숫자형 comm을 문자형으로 바꾸어준다.

102. 커미션이 null인 사원의 이름과 커미션을 출력하시오!

(is null 연산자 사용하지 말고 nvl함수로 수행하시오.)

```
select ename, comm
    from emp
    where nvl(comm, -1) = -1;
```

>>> comm이 '0' 인 사원이 있으니까 -1을 넣어준다!!

103. 이름, 월급, 직업, 보너스를 출력하는데 보너스가 직업이 SALESMAN 이면 4500을 출력하고 직업이 ANALYST면 2400을 출력하고 나머지 직업은 0을 출력하시오.

```
select ename, sal, job, decode(job, 'SALESMAN', 4500, 'ANALYST', 2400, 0) as 보너스
    from emp;
```

104. 이름, 입사한 년도 4자리로 출력하시오!

```
select ename, to_char(hiredate, 'RRRR')
    from emp;
```

105. 이름, 입사한 년도, 보너스를 출력하는데 보너스가 입사한 년도가 1980년이면 5000을 출력하고, 1981년이면 4000을 출력하고, 나머지 년도는 0으로 출력하시오!

```
select ename, to_char(hiredate, 'RRRR'), decode(to_char(hiredate, 'RRRR'),
                                                '1980', 5000,
                                                '1981', 4000, 0) as 보너스
    from emp;
```

106. 이름, 월급, 보너스를 출력하는데 보너스가 월급이 4000이상이면 500을 출력하고,
월급이 2000이상 4000이하면 300을 출력하고,
나머지 월급 사원들은 그냥 0을 출력하시오.

❖ 중요설명 : decode는 등호(=)비교만 가능하기 때문에 부등호 비교를 하려면 case문을 사용해야 한다.
case문은 등호비교, 부등호 비교 둘 다 가능합니다.

```
select ename, sal, case when sal >=4000 then 500
                       when sal >=2000 then 300   (자동으로 2000~4000 사이로 등록이 됨)
                       else 0 end
from emp;
```

>>>end로 꼭 마무리를 지어줄 것

107. 이름, 월급, 부서번호, 보너스를 출력하는데 보너스가 부서번호가 10번이면 500을 출력하고 부서번호가 20번이면 300을 출력하고 나머지 부서번호면 0을 출력하시오!

```
select ename, sal, deptno, case when deptno = 10 then 500
                                when deptno = 20 then 300
                                else 0 end as 보너스
    from emp;
```

108. 우리반 테이블에서 이름을 출력하고 그 옆에 보너스를 출력하는데
이름의 철자가 3글자이면 보너스를 7000을 출력하고
이름의 철자가 2글자이면 보너스를 5000을 출력하고
이름의 철자가 4글자이면 보너스를 4000을 출력하시오.

```
select ename, case when length(ename) >= 4 then 4000
                   when length(ename) >= 3 then 7000
                   else 5000 end as 보너스
    from emp12;
```

109. 우리반 테이블에서 이름 세글자로만 이름의 가운데 글제를 *로 출력하시오!

1. 이름의 철자의 갯수가 3글자와 2글자는 아래의 SQL로 수행

```
select replace(ename, substr(ename, 2, 1), '*')
    from emp12;
```

2. 이름의 철자의 갯수가 4글자면

```
select replace( ename, substr(ename, -2, 1), '*')
  from emp12;
```

110. 우리반 테이블의 이름의 철자의 갯수와 관계없이 일괄적으로 이름이 *이 아래와 같이 출력되게

하시오.

결과 : 남궁*미
허*
김*비
.
.

```
select decode(length(ename),3, replace(ename, substr(ename, 2, 1), '*'),
                               4, replace(ename, substr(ename, -2, 1), '*'),
                                  replace(ename, substr(ename, 2, 1), '*'))
   from emp12;
```

또는

```
select case when length(ename) >= 4 then replace(ename, substr(ename, -2,1), '*')
          else replace(ename, substr(ename, 2,1), '*') end
       from emp12;
```

111. emp(사원) 테이블에서 이름을 출력하고 입사한 요일을 출력하는데 입사한 요일이 월화수목금토일 순으로 출력하시오!

```
select ename,to_char(hiredate, 'day')
   from emp
   order by to_char(hiredate-1, 'd') asc;
```

112. 직업이 SALESMAN인 사원들의 최대월급을 출력하시오!

```
 select max(sal)
  from emp
  where job = 'SALESMAN';
```

113. 우리반에서 최소 나이인 학생의 나이를 출력하시오.

```
select min(age)
   from emp12;
```

114. 통신사가 sk인 학생들 중에서 최대 나이인 학생의 나이를 출력하시오!

```
select max(age)
   from emp12
   where lower(ltrim(telecom)) like 'sk%';
```

또는

```
select max(age)
from emp12
where lower(telecom) in ('sk', 'skt');
```

115. 30번 부서번호의 최대월급을 출력하시오!

```
select max(sal)
    from emp
    where deptno = 30;
```

>>> 2850

```
select deptno, max(sal)
    from emp
    where deptno = 30;
```

이렇게 하면 안나온다. max 셀은 딱 한개만 출력하려고 하기 때문에

>>>

```
select deptno, max(sal)
    from emp
    where deptno = 30
  group by deptno;
```

❖ 설명 : group by deptno를 하게 되면 여러 개 나오려는 deptno를 grouping 해준다.

116. 직업, 직업별 최대월급을 출력하는데 직업을 SALESMAN만 출력하시오.

```
select job, max(sal)
    from emp
    where job = 'SALESMAN'
    group by job;
```

117. (10/29) 직업이 ANALYST인 사원들의 최대월급을 출력하시오.

```
select max(sal)
    from emp
    where job = 'ANALYST';
```

118. 직업, 직업별 최대월급을 출력하시오!

```
select job, max(sal)        4
    from emp                1
    where job is not null   2
    group by job;           3 (실행순서)
```

119. 위의 결과를 출력하는데 직업별 최대월급이 높으것부터 출력하시오.

```
select job, max(sal)        4
    from emp                1
    where job is not null   2
```

```
group by job;                3
order by max(sal) desc;      5(실행순서)
```

120. 부서번호, 부서번호별 최대월급을 출력하는데 부서번호별 최대 월급이 높은 것부터 출력하시오!

```
select deptno, max(sal)
    from emp
    group by deptno
    order by max(sal) desc nulls last;
                           Δ
                           null이 맨 나중에 나오게 하는 것
```

❖ 설명 : order by 절의 옵션 : nulls last ---> null을 맨 뒤로 보내겠다.
nulls first ---> null을 맨 앞으로 보내겠다.

121. 이름과 연봉(sal*12)을 출력하는데 연봉을 출력할 때 천단위와 백만단위가 표시되게 하시오.

```
select ename, to_char(sal*12, '999,999,999')
    from emp;
```

122. 위의 결과를 다시 출력하는데 컬럼명을 한글로 연봉이라고 하고 연봉이 높은 사원부터 출력하시오!

```
select ename, to_char(sal*12, '999,999,999') as 연봉
    from emp
    order by to_char(sal*12, '999,999,999') desc;
```

123. 직업과 직업별 최대월급을 출력하는데 직업별 최대월급을 출력할 때, 천단위 콤마(,)가 출력되게 하시오.

```
select job, to_char(max(sal), '999,999')
    from emp
    group by job;

select job, max(to_char(sal, '999,999')
```

라고 해도 되지만, 문자중에서 최대를 뽑는것 보다 숫자중에서 뽑는게 나음.

124. 직업, 직업별 최소월급을 출력하시오.

```
select job, min(sal)
    from emp
    where job is not null
    group by job;
```

125. 위의 결과를 다시 출력하는데 직업이 ABCD순서대로 출력하게 하시오.

```
select job, min(sal)
    from emp
    group by job
    order  by job asc;
```

126. 위의 결과에서 직업이 SALESMAN은 제외하고 출력하시오!

```
select job, min(sal)
    from emp
    where job != 'SALESMAN'
    group by job
    order by job asc;
```

>>> null도 안나오고, SALESMAN도 안나옴. where 절 값을 비교할 수 없기 때문에 null이 안 나옴.

127. 우리반에서 최소나이를 출력하시오.

```
select min(age)
    from emp12;
```

❖ 설명 : 만약 이름도 같이 출력하려면 지금까지 배운 내용으로는 안되고 서브쿼리를 사용해야 합니다.

128. 서울에서 사는 학생중에 최소나이를 출력하시오.

```
select min(age)
    from emp12
    where address like '서울%';
```

❖ 설명 : 와일드 카드를 양쪽에 사용하게 되면 검색 성능이 느려지게 됩니다. 와일드 카드가 뒤쪽에 있는 경우는 상관 없으나 앞에 있으면 검색성능이 느려집니다.

129. 입사한 년도(4자리)를 출력하고 입사한 년도별 최소월급을 출력하시오!

```
select to_char(hiredate, 'RRRR'), min(sal)
    from emp
    group by to_char(hiredate, 'RRRR');
```

130. 우리반 나이의 평균값을 출력하세요!

```
select avg(age)
    from emp12;
```

131. 사원 테이블의 월급의 평균값을 출력하시오!

```
select avg(sal)
    from emp;
```

설명 : 월급을 다 더한 후 전체 인원수인 14로 나눔 (전체인원 수)

132. 커미션의 평균값을 출력하시오

```
select avg(comm)
    from emp;
```

>>> 550

설명 : 커미션을 다 더한 후 4로 나눔. "그룹함수는 null값을 무시한다"

133. 위의 결과를 다시 출력하는데 4로 나누지 않고 14로 나누게 하시오.

```
select avg(nvl(comm,0))
    from emp;
>>>146.666667
```

❖ 설명 : null값을 0으로 다 변경했기 때문에 4로 나누지 않고 14로 나눔

134. 위의 결과에서 소숫점 이하는 안나오게 반올림 하시오!

```
select round(avg(nvl(comm,0)))
    from emp;
```

135. 직업, 직업별 평균월급을 출력하시오.

```
select job, avg(sal)
    from emp
    where job is not null
    group by job;
```

136. 통신사, 통신사별 평균나이를 출력하시오!

```
select lower(substr(telecom,1,2)), avg(age)
    from emp12
    group by lower(substr(telecom,1,2));
```

또는

```
select decode(lower(telecom), 'skt', 'sk', lower(telecom)),
        avg(age)
        from emp12
        group by decode(lower(telecom), 'skt', 'sk', lower(telecom));
```

❖ 설명 : decode(lower(telecom), 'skt', 'sk', lower(telecom))의 뜻은 lower(telecom) 데이터가 skt면 sk로 출력하고 나머지 통신사는 그냥 그대로 lower(telecom)으로 출력해라 라는 뜻

137. 전공을 출력하고, 전공별 평균나이를 출력하는데 전공이 ㄱ,ㄴ,ㄷ,ㄹ순으로 출력되게 하시오.

```
select decode(major, '통게학과', '통계학과', major), avg(age)
    from emp12
    group by major
    order by major asc;
```

138. 직업, 직업별 토탈월급을 출력하는데 직업별 토탈월급이 높은것부터 출력하시오.

```
select job, sum(sal)          3
    from emp                  1
    group by job              2
    order by sum(sal) desc;   4
```

139. 위의 결과에서 직업이 SALESMAN은 제외하고 출력하시오!

```
select job, sum(sal) as 토탈        4
    from emp                        1
    where job != 'SALESMAN'         2
    group by job                    3
    order by sum(sal) desc;         5
```

140. 위의 결과를 다시 출력하는데 토탈 월급이 6000이상인 것만 출력하시오

```
select job, sum(sal) as 토탈
    from emp
    where job !='SALESMAN'
    group by job
    having sum(sal)>=6000
    order by sum(sal) desc;
```

❖ 설명 : group 함수로 조건을 줄 때는 where절에 사용하면 안되고 having절에 사용해야 합니다.
where절에는 그룹함수를 사용하지 않은 일반적인 검색조건을 줄 때 사용합니다.

select문의 6가지 절
코딩순서 : select --> from --> where --> group by --> having --> order by
실행순서 : from --> where --> group by --> having --> select --> order by

141. 직업, 직업별 토탈월급을 출력하는데 직업이 SALESMAN은 제외하고 출력하고 직업별 토탈월급이 6000이상인것만 출력하고 직업별 토탈월급이 높은 것부터 출력하시오.

```
select job, sum(sal) as 토탈월급
    from emp
    where job != 'SALESMAN'
    group by job
    having sum(sal) >= 6000
    order by sum(sal) desc;
----------
select job, sum(sal)as 토탈월급
    from emp
    group by job
    having sum(sal) >= 6000 and job != 'SALESMAN'   <<< 일반적인 조건
    order by sum(sal) desc;
```

❖ 이것도 실행이 된다. 그러나 이렇게 작성하면 검색속도가 느려지므로 이렇게 작성하면 안된다.
❖ 그룹함수가 아닌 일반적인 검색조건을 having절에 사용하면 검색속도가 느려지므로 반드시 where절에 작성해야합니다.

142. 통신사, 통신사별 토탈나이를 출력하는데 skt는 제외하고 출력하고 통신사별 토탈나이가 100살

이상인 데이터만 출력하고 통신사별 토탈나이가 높은 것부터 출력하시오.

```
select lower(telecom), sum(age)
    from emp12
    where lower(telecom) != 'skt'
    group by lower(telecom)
    having sum(age) >=100
    order by sum(age) desc;
```

select 절과 group by절에 lower을 써주지 않으면 값이 달라질 수 있다.
그래서 lower을 꼭 써주어야 한다.

143. 위의 문제를 다시 푸는데 이번에는 skt를 sk에 포함시켜서 출력하시오!

```
select decode(lower(telecom), 'skt', 'sk', lower(telecom))as 통신사, sum(age)
    from emp12
    group by decode(lower(telecom), 'skt', 'sk', lower(telecom))
    having sum(age) >=100
    order by sum(age) desc;
```

또는

```
select lower(substr(telecom, 1, 2)), sum(age)
   from emp12
   group by lower(substr(telecom, 1, 2))
   having sum(age) >= 100
   order by 2 desc;
```

144. 입사한 년도(4자리), 입사한 년도별 토탈월급을 출력하는데 토탈월급을 출력할 때 천단위 표시가 출력되게 하시오.

```
select to_char(hiredate, 'RRRR'), to_char(sum(sal),'999,999')
    from emp
    group by to_char(hiredate, 'RRRR');
```

145. 직업, 직업별 인원수를 출력하시오!

```
select job, count(*) ☜보통 여기에 부모키 컬럼을 씀
    from emp
    group by job;
```

설명 :
```
select job, count(empno)
        from emp
        group by job;
```

그러나 부모키 컬럼을 잘 모르는 경우가 있고, null값이 있을 경우를 대비해서 *을 씀

146. 우리반 나이, 나이별 인원수를 출력하시오!

```
select age, count(*)
    from emp12
```

```
group by age;
```

147. 위의 결과를 다시 출력하는데 나이별 인원수가 높은 것부터 출력하시오!

```
select age, count(*)
    from emp12
    group by age
    order by count(*) desc;
```

148. 위의 결과를 다시 출력하는데 나이별 인원수가 2명이상인 것만 출력하시오.

```
select age, count(*)
    from emp12
    group by age
    having count(*) >= 2
    order by count(*) desc;
```

149. 통신사, 통신사별 인원수를 출력하시오!

```
select lower(substr(telecom,1,2)), count(telecom)
    from emp12
    group by lower(substr(telecom,1,2));
```

또는

```
select decode(lower(telecom), 'skt','sk', lower(telecom), count(telecom)
    from emp12
    group by decode(lower(telecom), 'skt','sk', lower(telecom);
```

150. 이름, 이메일의 도메인만 출력하시오!

구윤모 naver

권세원 gmail

.

.

.

151. 이메일 도메인, 이메일 도메인별 인원수를 출력하시오!

```
select substr(email, instr(email, '@')+1, instr(substr(email,instr(email,'@')), '.')-2) as 도메인,
        count (substr(email, instr(email, '@')+1, instr(substr(email,instr(email,'@')), '.')-2))
    from emp12
    group by substr(email, instr(email, '@')+1, instr(substr(email,instr(email,'@')), '.')-2);
```

152. 주소, 주소별 인원수를 아래와 같이 출력하시오!

서울시 19

경기도 1

.

.

.

```
select substr(address,1,3) , count(address)
    from emp12
    group by substr(address,1,3);
```

153. 이름, 나이, 순위를 출력하는데 순위가 나이가 높은 순서대로 순위를 출력하시오.

```
select ename, age, rank () over (order by age desc) as 순위
    from emp12;
```

154. 직업, 이름, 월급, 순위를 출력하는데 직업별로 각각 월급이 높은 순서대로 순위를 출력되게 하시오!

```
select job, ename, sal, rank () over (partition by job
                                      order by sal desc) as 순위
    from emp;
```

❖ 설명 : partition by 는 group by 하고 혼동하면 안된다.
partition by는 데이터 분석함수에서 괄호 안에 쓰는 문법으로
partition by job이라고 하면 직업별로 각각 파티션해서 나누겠다는 것이다.
그래서 아래의 뜻은
rank () over (partition by job <<< 직업별로 파티션해서
order by sal desc) <<< 월급이 높은 순서대로 순위를 출력하겠다.

155. 부서번호, 이름, 월급, 입사일, 순위를 출력하는데 순위가 부서번호별로 각각 먼저 입사한 사원 순으로 순위가 부여되게 하시오!

```
select deptno, ename, hiredate, rank () over (partition by deptno
                                              order by hiredate asc) as 순위
    from emp;
```

156. 통신사, 이름, 나이, 순위를 출력하는데 통신사별로 각각 나이가 높은 학생순으로 순위를 부여하시오.

```
select lower(substr(telecom,1,2)), ename, age,
                                    rank () over (partition by lower(substr(telecom,1,2))
                                    order by age desc)
    from emp12;
```

157. 이름, 월급, 순위를 출력하는데 dense_rank를 써서 순위가 1,2,3,4,5...등 전부 출력되도록 하시오. 순위는 월급이 높은 순서에 대한 순위입니다.

```
select ename, sal, dense_rank() over( order by sal desc)
    from emp;
```

158. 입사한 년도(4자리), 이름, 월급, 순위를 출력하는데 순위가 입사한 년도별로 각각 월급이 높은 순서대로 순위를 출력하시오.

```
lect to_char(hiredate, 'RRRR'), ename, sal,
                                        dense_rank()over (partition by to_char(hiredate, 'RRRR')
                                        order by sal desc)
        from emp;
```

❖ 월급이 같아서 같은 순위가 여러 개 나오면 다음 순위가 그 다음 순위가 중복된 수만큼 더해서 출력되는 것이 rank()이고 중복되었더라도 그냥 바로 그 다음 순위가 나오는게 dense_rank()이다

159. 이메일 도메인, 이름, 나이, 순위를 출력하는데 순위가 이메일 도메인 별로 각각 나이가 높은 학생 순으로 출력되게 하시오!

```
select substr(email, instr(email, '@')+1, instr(substr(email,instr(email,'@')), '.')-2)as 도메인,
ename, age,
        dense_rank () over (partition by substr(email, instr(email, '@')+1,
instr(substr(email,instr(email,'@')), '.')-2)
                        order by age desc)as 순위
    from emp12;
```

160. 이름, 입사일, 순위를 출력하는데 순위가 먼저 입사한 사원순으로 순위를 부여하시오.

```
select ename, hiredate, rank () over (order by hiredate asc) 순위
    from emp;
```

161. 직업, 이름, 입사일, 순위를 출력하는데 순위가 직업별로 각각 먼저 입사한 사원 순으로 순위를 부여하시오.

```
select job, ename, hiredate, rank () over (partition by job
                                        order by hiredate asc)순위
    from emp;
```

162. 이름, 월급, 순위를 출력하는데 순위가 월급이 높은 순서대로 순위를 부여하시오!

```
select ename, sal, dense_rank()over (order by sal desc)순위
    from emp;
```

(업무나 상황에 따라 rank 또는 dense_rank로 쓰면 됨)

163. 월급이 2975는 순위가 몇위인가?

```
select dense_rank(2975) within group(order by sal desc) 순위
from emp;
```

❖ 설명 : within ~이내에
월급이 높은 순서대로 정렬한 그룹 안에서 몇위의 순위인가? 라는 뜻

164. 우리반에서 34나이의 순위를 출력하시오!

```
select dense_rank(34) within group(order by age desc) 순위
    from emp12;
```

165. 81년 11월 17에 입사한 사원은 사원테이블에서 몇번째로 입사한 사원인가?

```
select dense_rank(to_date('81/11/17', 'RR/MM'DD') within group(order by hiredate asc)순위
    from emp;
```
>>>이렇게 to_date를 써줘야 에러안나고 정확하게 데이터를 얻을 수 있음

166. 이름, 나이, 등급을 출력하는데 등급을 7등급으로 나눠서 출력하시오. (나이가 높은 순서대로 등급을 나누시오.)

```
select ename, age, ntile(7) over (order by age desc)등급
    from emp12;
```

167. 직업, 이름, 월급, 등급을 출력하는데 직업별 각각 등급이 3등급으로 나눠지게 하시오! (등급은 월급이 높은 순서대로의 등급입니다.)

```
select job, ename, sal, ntile(3) over (partition by job
                                       order by sal desc)등급
    from emp;
```

168. 다음 결과에서 소수점 세번째까지만 출력되게끔 반올림 하시오!

```
select ename, sal,
            cume_dist() over (order by sal desc) 비율
      from emp;
>>>
select ename, sal,
            round(cume_dist() over (order by sal desc), 3) 비율
from emp;
```

169. 직업, 직업별로 해당하는 사원들의 이름을 가로로 출력하시오.

```
select job, listagg(ename, ', ') within group (order by ename asc) 이름
    from emp
    group by job;
```

170. 아래와 같이 결과를 출력하시오 (이름 옆에 월급이 같이 나오게)

ANALYST FORD(3000), SCOTT(3000)
CLERK ADAMS(1100), JAMES(950)

```
select job, listagg(ename||'('||sal||')',', ') within group (order by ename asc) 이름
    from emp
    group by job;
```

171. 나이, 나이별로 해당하는 학생들의 이름을 가로로 출력하시오.

```
select age, listagg(ename, ', ') within group (order by ename asc) 이름
    from emp12
    group by age;
```

172. 통신사를 출력하고 통신사별로 해당하는 학생들의 이름을 출력하는데 이름 옆에 나이도 같이 출력되게 하고 나이가 높은 학생순으로 출력되게 하시오

```
select telecom, listagg(ename||'('||age||')', ', ')
                within group (order by age desc)
    from emp12
    group by telecom;
```

173. 이름, 입사일, 바로 전에 입사한 사원의 입사일, 바로 다음에 입사한 사원의 입사일을 출력하시오!

```
select ename, hiredate, lag(hiredate, 1) over (order by hiredate asc) as 전입사자,
                        lead(hiredate, 1) over (order by hiredate asc) as 다음입사자
                from emp;
```

174. 통신사, 통신사별 토탈 나이를 출력하시오 (세로 출력)

```
select telecom, sum(age)
    from emp12
    group by telecom;
```

175. 이번에는 아래와 같이 가로로 출력하시오!

```
sk      lg      kt
322    126    411
```

```
select sum(decode(telecom, 'sk', age, 0))as "sk", sum(decode(telecom, 'kt', age, 0)) as "kt",
sum(decode(telecom, 'lg', age, 0)) as "lg"
    from emp12;
```

176. 아래의 SQL 두개는 결과가 같을까?

"그룹함수는 null값을 무시한다."

select sum(comm) from emp; ---> null값이 아닌 4건만 다 더했다.
>>>2200

select sum(nvl(comm,0)) from emp; ---> null을 0으로 변경하고 sum할 때 0을 연산에 포함
>>>2200

같다

❖ 설명 : 위의 SQL이 성능이 더 좋다. 왜냐하면 null값은 sum연산에 포함되지 않기 때문이다.

177. 아래의 SQL을 튜닝하시오.

튜닝 전:

```
select sum(decode(telecom, 'sk', age, 0))as "sk",
       sum(decode(telecom, 'kt', age, 0)) as "kt",
       sum(decode(telecom, 'lg', age, 0)) as "lg"
    from emp12;
```

튜닝 후:

```
select sum(decode(telecom, 'sk', age, null))as "sk",
            sum(decode(telecom, 'kt', age, null)) as "kt",
            sum(decode(telecom, 'lg', age, null)) as "lg"
```

```
from emp12;
```

178. 직업, 직업별 토탈월급을 출력하시오 (세로로 출력)

```
select job, sum(sal)
    from emp
    group by job;
```

179. 직업, 직업별 토탈월급을 출력하시오 (가로로 출력)

```
select sum(decode(job, 'SALESMAN', sal, null)) as "SALESMAN",
       sum(decode(job, 'CLERK', sal, null)) as "CLERK",
       sum(decode(job, 'ANALYST', sal, null)) as "ANALYST",
       sum(decode(job, 'MANAGER', sal, null)) as "MANAGER",
       sum(decode(job, 'PRESIDENT', sal, null)) as "PRESIDENT"
    from emp;
```

180. 통신사, 통신사별 토탈나이를 가로로 출력하시오 (pivot문으로)

```
select *
    from (select telecom, age from emp12) <<< 여기는 그룹함수를 쓰지 않는다(이 문제에서는)
    pivot( sum(age) for telecom in ('sk', 'kt', 'lg'));


select *
    from (select telecom, age from emp12)
    pivot( sum(age) for telecom in ('sk' as "sk", 'kt' as "kt", 'lg' as "lg"));
                                        Δ 이렇게 해주면 컬럼명에 ''가 붙지 않는다
```

181. 위의 결과를 토탈나이가 아니라 평균나이로 나오게 하시오.

```
select *
    from (select telecom, age from emp12)
    pivot( avg(age) for telecom in ('sk' as "sk", 'kt' as "kt", 'lg' as "lg"));
```

182. 이름, 나이, 나이의 누적치를 출력하시오. (이름을 ㄱㄴㄷㄹ순으로 정렬)

```
select ename, age, sum(age) over(order by ename asc) 누적치
    from emp12;
```

183. 직업, 이름, 월급의 누적치를 출력하는데 직업별로 각각 월급의 누적치가 출력되게 하시오.

```
select job, ename, sal, sum(sal) over (partition by job
                                       order by sal asc) 누적치
    from emp;
```

184. 통신사, 이름, 나이, 나이의 누적치를 출력하는데 나이의 누적치가 통신사별로 각각 누적되어서 출력되게 하시오!

```
select telecom, ename, age, sum(age) over (partition by telecom
                                           order by age asc)
                                           from emp12;
```

185. 통신사, 통신사별 토탈나이를 출력하는데 맨 위에 전체 토탈나이가 출력되게 하시오.

```
select telecom, sum(age)
    from emp12
    group by cube(telecom);
```

186. 입사한 년도(4자리), 입사한 년도별 토탈월급을 출력하는데 맨 위에 전체토탈 월급을 출력하시오.

```
select to_char(hiredate, 'RRRR') , sum(sal)
    from emp
    group by cube(to_char(hiredate, 'RRRR'));
```

187. 입사한 년도(4자리), 입사한 년도별 토탈월급을 출력하시오!

1. 세로출력
2. 가로출력 (sum + decode 사용)

1. 세로 출력

```
select to_char(hiredate, 'RRRR') , sum(sal)
    from emp
    group by to_char(hiredate, 'RRRR');
```

2. 가로 출력(sum + decode 사용)

```
select sum(decode(to_char(hiredate, 'RRRR'), '1980',sal,null)) as "1980",
        sum(decode(to_char(hiredate, 'RRRR'), '1981', sal,null)) as "1981",
        sum(decode(to_char(hiredate, 'RRRR'), '1982', sal,null)) as "1982",
        sum(decode(to_char(hiredate, 'RRRR'), '1983', sal,null)) as "1983"
     from emp;
```

188. 직업, 직업별 토탈월급을 출력하시오!

```
select job, sum(sal)
    from emp
    group by job;
```

189. 그러면 위의 결과를 다시 출력하는데 맨 아래에 전체 토탈 월급이 출력되게 하시오

```
select job, sum(sal)
    from emp
    group by rollup(job);
```

190. 아래의 job 맨아래 null로 나오는 부분에 토탈값이르고 한글로 null대신 출력되게 하시오.

```
select nvl(job, '토탈값'), sum(sal)
    from emp
    group by rollup(job);
```

191. 위의 결과를 다시 출력하는데 컬럼명이 아래와 같이 출력되게 하시오!

JOB SUM(SAL)
---------- ----------
ANALYST 6000
CLERK 4150

MANAGER 8275
PRESIDENT 5000
SALESMAN 5600
토탈값 29025

```
select nvl(job, '토탈값')as JOB, sum(sal)
    from emp
    group by rollup(job);
```

192. 위의 결과를 다시 출력하는데 아래와 같이 토탈월급 부분에 천단위를 부여하시오!

```
select nvl(job, '토탈값')as JOB, to_char(sum(sal), '999,999')
    from emp
    group by rollup(job);
```

193. 직업, 직업별 토탈월급을 출력하는데 맨 아래에 전체 토탈월급이 출력되게 하시오!
(grouping sets를 이용해서 수행)

```
select job, sum(sal)
    from emp
    group by grouping sets(job, ());
```

194. 부서번호와 직업을 출력하고 그 옆에 부서번호별 직업별 토탈월급을 출력하시오.

```
select deptno, job, sum(sal)
    from emp
    group by deptno, job;
```

❖ 설명 : select절에 그룹함수와 함께 나열한 컬럼들은 반드시 group by 절에 명시 해줘야 에러가 나지 않고 출력될 수 있다.

195. 부서번호와 직업을 출력하고 그 옆에 부서번호별 직업별 토탈 월급을 출력하고 동시에 부서번호별 토탈월급도 중간 중간 출력되게 하시오.

```
select deptno, job, sum(sal)
    from emp
    group by grouping sets((deptno, job), (deptno));
```

196. 위의 결과의 맨 아래에 전체 토탈월급이 출력되게 하시오!

```
select deptno, job, sum(sal)
    from emp
    group by grouping sets((deptno, job), (deptno), ());
```

Δ전체를 뽑겠다.

여기서 (deptno), ()는 뺄 수 있지만 (deptno, job)은 제거하지 못한다.

197. 사원번호, 사원 이름, 월급을 출력하는데 맨 아래에 전체토탈월급을 출력하시오.

```
select empno, ename, sum(sal)
    from emp
```

```
group by grouping sets ((empno,ename), ());
```

❖ 설명 : 사원번호는 중복되지 않기 때문에 grouping sets함수의 괄호 안에 grouping결과로 넣어도 자기 월급이 출력되게 된다. 왜냐하면 사원번호는 딱 한건이기 때문이다. 그 한건을 집계해 보면 그 값이 된다. 그래서 자신있게 (empno, ename)이라고 작성한 것이다.
그리고 맨 끝의 ()은 전체 토탈월급을 출력하는 것이다.

```
select empno, ename, sal
    from emp
    group by grouping sets ((empno,ename,sal),());
```

이렇게 해도될까? 정답 : NO

그룹함수와 grouping sets는 서로 짝꿍이라서 grouping sets를 썼는데 select 절에 group함수를 안쓰면 안된다.

198. 우리반 테이블에서 통신사, 이름, 나이를 출력하는데 중간중간 통신사별 토탈나이가 출력되게 하시오.

```
select telecom, ename, sum(age)
    from emp12
    group by grouping sets((telecom, ename), (telecom), ());
```

199. 입사한 년도(4자리), 입사한 년도별 토탈월급을 출력하는데 입사한 년도별 전체 토탈월급이 맨 아래에 출력되게 하시오!

```
select to_char(hiredate, 'RRRR'), sum(sal)
    from emp
    group by grouping sets(to_char(hiredate, 'RRRR'), ());
```

또는

```
select to_char(hiredate, 'RRRR') sum(sal)
    from emp
   group by rollup(to_char(hiredate, 'RRRR'));
```

200. 위의 결과를 다시 출력하는데 아래와 같이 천단위를 부여해서 출력하시오.

```
select to_char(hiredate, 'RRRR'), to_char(sum(sal), '999,999')
    from emp
    group by grouping sets(to_char(hiredate, 'RRRR'), ());
```

201. 입사한 년도 (4자리), 부서번호, 입사한 년도별 부서번호별 토탈월급을 출력하시오.

```
select to_char(hiredate, 'RRRR'), deptno, sum(sal)
    from emp
    group by to_char(hiredate, 'RRRR'), deptno;
```

202. 입사한 년도 (4자리), 부서번호, 입사한 년도별 부서번호별 토탈월급을 출력하는데 중간중간 입사

한 년도(4자리)별 토탈월급이 출력되게 하고 맨 아래에 전체 토탈월급이 출력되게 하시오.

```
select to_char(hiredate, 'RRRR'), deptno, sum(sal)
from emp
group by grouping sets ((to_char(hiredate, 'RRRR'), deptno), (to_char(hiredate, 'RRRR')), ( ));
```

203. 직업이 SALESMAN인 사원들의 이름과 직업을 출력하는데 row_number 함수를 사용해서 맨 앞에 번호 차례대로 부여되게 하시오.

```
select row_number() over (order by ename asc)번호, ename, job
    from emp
    where job = 'SALESMAN';
```

❖ 설명 : order by 다음에 empno를 써도 되고 ename을 써도 되고 정렬하고 싶은대로 컬럼을 기술하면 된다. order by empno(정렬할 대상) 다음에 asc를 안쓰면 기본 값이 asc이다.

204. 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하는데 맨 앞에 row_number () 함수를 써서 번호가 부여되게 하시오.
(월급이 높은순서대로 출력되면서 번호가 부여되게 하시오.)

```
select  row_number()  over  (order by sal desc) 번호,
        ename, sal, job
 from  emp
 where  job='SALESMAN';
```

❖ 설명 : 위의 결과에서 번호 1번만 출력하는것은 인라인뷰를 배워야 할 수 있습니다.

205. 직업이 SALESMAN인 사원들의 데이터 중 2개만 출력하시오!
(사원이름, 직업과 월급을 출력하시오)

```
select rownum, ename, job, sal
    from emp
    where rownum <=2 and job = 'SALESMAN';
```

206. 위의 결과에서 한건만 출력되게 하시오!

```
select rownum, ename, job, sal
    from emp
    where rownum <=1 and job = 'SALESMAN';
```

아래의 SQL의 결과는 없습니다.

```
select rownum, ename, job, sal
    from emp
    where rownum = 2 and job = 'SALESMAN';
```

왜냐하면? rownum 쓸 때는 부등호 비교와 같이 사용해야 합니다.
등호 비교는 숫자 1만 비교 됩니다.

>>>>

```
select rownum, ename, job, sal
    from emp
    where rownum = 1 and job = 'SALESMAN';
```

>>> 이건 결과가 나옵니다.

만약에 우리 건물앞에 강남역을 가는 미니버스가 있는데 앞에 광장의 사람들에게 이 버스에 탈 사람중 제일 먼저 탈 사람은 타세요~ 라고하면 제일 먼저 달려간 사람이 타면 된다. 그런데 이 버스에 2번째로 탈 사람 타세요~ 라고 하면 다 머뭇거리고 못 탈 것 입니다.
누군가 한명이 먼저 가야 두번째 사람이 갈 수 있습니다.

```
select rownum, ename, job, sal
    from emp
    where rownum between 2 and 5;
```

값이 나올까? NO

❖ 설명 : 실제로 위의 SQL의 결과를 보려면 인라인뷰를 사용해야 합니다.

207. 우리반 테이블의 데이터를 가져오는데 위의 3건만 가져와서 출력하시오!

```
select *
    from emp12
    where rownum <=3;
```

208. 우리반에서 나이가 가장 많은 학생들의 이름과 나이를 출력하는데 한 5명만 출력하시오!

```
select ename, age
    from emp12
    order by age desc
    fetch first 5 rows only;
```

209. 직업, 직업별 토탈월급을 출력하는데 직업별 토탈월급이 높은것부터 출력하고 위쪽에 2개의 행만 출력하시오!

```
select job, sum(sal)
    from emp
    group by job
    order by sum(sal) desc
    fetch first 2 rows only; <<<< fetch는 맨 나중에!!
```

210. 직업이 SALESMAN인 사원들의 이름과 월급과 직업과 부서위치를 출력하시오!

```
select ename, sal, job, loc
    from emp, dept
    where emp.deptno = dept.deptno and                    job = 'SALESMAN';
```

Δ 조인조건 : 두개의 테이블이 서로 연관이 있다라는 조건 Δ 검색조건

211. 월급이 2000이상인 사원들의 이름과 월급과 부서위치를 출력하시오!

```
select ename, sal, loc
    from emp, dept
    where emp.deptno = dept.deptno and sal >= 2000;
```

212. 위의 결과에서 이름, 월급, 부서위치 옆에 부서번호도 같이 출력하시오!

```
 select ename, sal, loc, deptno
      from emp, dept
      where emp.deptno = dept.deptno and sal >= 2000;
select ename, sal, loc, deptno
                    *
1행에 오류:
ORA-00918: 열의 정의가 애매합니다
>>>
```

이렇게 하면 에러남

```
select ename, sal, loc, emp.deptno
    from emp, dept
    where emp.deptno = dept.deptno and sal >= 2000;
```

- 설명: emp.deptno라고 작성함으로써 emp테이블에 있는 부서번호를 가져와라~ 라고 해줘야 합니다.
- 에러가 나지 않는다고 컬럼명 앞에 테이블명을 적지 않는 습관을 버리고 무조건 조인문장에서는 컬럼명 앞에 테이블명을 적어줍니다.
  테이블.컬럼명 으로 작성해줘야 검색속도가 더 빨라집니다. (안 뒤져봐도 돼서..)

```
select emp.ename, emp.sal, dept.loc, emp.deptno
    from emp, dept
    where emp.deptno = dept.deptno and sal >= 2000;
```

↓코드를 좀 더 간결하게 작성

```
select e.ename, e.sal, d.loc, e.deptno
      from emp e, dept d
      where e.deptno = d.deptno and e.sal >=2000;
```

※ 설명 : emp는 e라고 하고 dept는 d라고 별칭을 주어서 별칭을 사용해서 코딩을 하면 좀 더 간결하게 조인문장을 작성할 수 있습니다.

```
select emp.ename, e.sal, d.loc, e.deptno
      from emp e, dept d
      where e.deptno = d.deptno and e.sal >=2000;
```

※ 설명 : 위의 문장은 에러가 납니다. 왜냐하면 emp는 e로 변경되었기 때문입니다.

213. 월급이 1000에서 3000 사이인 사원들의 이름과 월급과 부서위치를 출력하시오.

```
select e.ename, e.sal, d.loc
    from emp e, dept d
    where e.deptno = d.deptno and e.sal between 1000 and 3000;
```

214. 사원번호가 7788, 7902, 7369번인 사원의 사원번호와 이름과 월급과 부서위치를 출력하시오!

```
select e.ename, e.sal, d.loc
    from emp e, dept d
    where e.deptno = d.deptno and e.empno in (7788, 7902, 7369);
```

※ 설명 : 하나의 값을 비교할 때는 =을 사용하지만 여러개의 값을 검색해서 비교하려면 in을 사용해야 합니다.

215. 이름의 첫번째 철자가 S로 시작하는 사원의 이름과 월급과 부서위치를 출력하시오!

```
select e.ename, e.sal, d.loc
    from emp e, dept d
    where e.deptno = d.deptno and e.ename like 'S%';
```

조인조건 ↓ e.deptno = d.deptno

검색조건 ↓ e.ename like 'S%'

※ 설명: 두개 이상의 테이블을 조인(합치기)을 하려면 반드시 조인조건을 where절에 기술해야 합니다.

216. DALLAS에서 근무하는 사원들의 이름과 직업과 부서위치를 출력하시오.

```
select e.ename, e.job, d.loc
    from emp e, dept d
    where e.deptno = d.deptno and d.loc = 'DALLAS';
```

217. 부서위치, 부서위치별 토탈월급을 출력하시오.

```
select d.loc, sum(e.sal)
    from emp e, dept d
    where e.dept = d.dept
    group by d.loc;
```

218. 부서위치, 부서위치 별 평균월급을 출력하는데 소수점 이하는 안나오게 반올림을 하고 부서위치별 평균월급이 높은 것부터 출력하고 부서위치별 평균월급이 출력될 때에 천단위를 부여하시오.

```
select d.loc, to_char(round(avg(e.sal)), '999,999')
    from emp e, dept d
    where e.deptno = d.deptno
    group by d.loc
    order by to_char(round(avg(e.sal)), '999,999') desc;
```

219. 월급이 2700이상인 사원들의 이름과 월급과 부서위치를 출력하시오.

```
select e.ename, e.sal, d.loc
    from emp e, dept d
    where e.deptno= d.deptno and e.sal >= 2700;
```

※ 설명 : 테이블 별칭을 사용해서 SQL코딩을 좀 심플하게 작성하세요.
그리고 반드시 조인문장을 작성할 때 컬럼명 옆에 테이블 별칭을 사용하세요.

1. 검색속도 빨라진다.
2. SQL 유지보수가 쉬워진다. (가독성)

220. 이름의 끝 글자가 t로 끝나는 사원들의 이름과 월급과 부서위치와 부서명을 출력하시오.

```
select e.ename, e.sal, d.loc, d.dname
    from emp e, dept d
    where d.deptno = e.deptno and e.ename like '%T';
```

221. 직업이 SALESMAN이고 월급이 1200 이상인 사원들의 이름과 직업과 부서위치와 월급을 출력하시오!

```
select e.ename, e.job, d.loc, e.sal
    from emp e, dept d
    where d.deptno = e.deptno and e.job = 'SALESMAN'
                                and e.sal>= 1200;
```

222. 부서위치, 부서위치별 토탈월급을 출력하는데 DALLAS는 제외하고 출력하시오!

```
select d.loc, sum(e.sal)
    from emp e, dept d
    where d.deptno = e.deptno and d.loc != 'DALLAS'
    group by d.loc;
```

223. 지금 출력된 결과를 다시 출력하는데 토탈월급이 높은것부터 출력하시오!

```
select d.loc, sum(e.sal)
    from emp e, dept d
    where d.deptno = e.deptno and d.loc != 'DALLAS'
    group by d.loc
    order by sum(e.sal) desc;
```

224. 다음 결과에서 등급이 3등급인 사람들만 출력하시오!

```
select e.ename, e.sal, s.grade
    from emp e, salgrade s
    where e.sal between s.losal and s.hisal;
```

>>>
정답:

```
select e.ename, e.sal, s.grade
    from emp e, salgrade s
    where e.sal between s.losal and s.hisal and s.grade = 3;
```

225. 급여등급 (grade), 급여등급별로 해당하는 사원들의 이름을 가로로 출력하시오.

grade ename
1 ADAMS, JAMES, SMITH
2
3
4
5 KING

```
select s.grade, listagg(e.ename, ', ') within group(order by e.ename asc) 이름
    from emp e, salgrade s
    where e.sal between s.losal and s.hisal
    group by s.grade;
```

226. 위의 결과에서 월급도 옆에 같이 나오게 하시오.

보기:

1 ADAMS(1100), JAMES(950), SMITH(800)
2 MARTIN(1250), MILLER(1300), WARD(1250)
3 ALLEN(1600), TURNER(1500)
4 BLAKE(2850), CLARK(2450), FORD(3000), JONES(2975), SCOTT(3000)
5 KING(5000), jack(3500)

정답:

```
select s.grade, listagg(e.ename||'('||e.sal||')', ', ') within group(order by e.ename asc) 이름
        from emp e, salgrade s
        where e.sal between s.losal and s.hisal
        group by s.grade;
```

227. emp와 dept를 서로 조인해서 이름과 부서위치와 부서번호를 출력하시오!

```
select e.ename, d.loc
    from emp e, dept d
    where e.deptno = d.deptno;
```

>>> 부서번호 40번이 조인이 안됨.

※ 설명 : 위의 결과를 보면 부서번호 40번은 조인이 되지 않아서 결과가 출력되지 않았습니다.
왜 조인이 안됐나요? 사원 테이블에 40번 부서번호가 없어서 입니다.

```
select e.ename, d.loc
      from emp e, dept d
      where e.deptno(+) = d.deptno;
                        아우터조인 싸인(모자란쪽에 붙여줌)
```

228. 부서위치, 부서위치별 토탈월급을 출력하시오!

```
select d.loc, sum(e.sal)
    from emp e, dept d
    where e.deptno= d.deptno
    group by d.loc;
```

❖ 설명 : 위의 조인된 결과를 보면 부서위치(loc)쪽에 BOSTON이 안 보입니다.

229. 위의 결과에서 BOSTON도 나오도록 조인문법을 변경하세요.

```
select d.loc, sum(e.sal)
    from emp e, dept d
    where e.deptno(+) = d.deptno
    group by d.loc;
```

230. 우리반 테이블과 telecom_price 테이블을 조인해서 이름, 나이, 성별, 통신사, 통신사 기본요금(price)을 출력하는데 나이가 27 이상인 학생들만 출력되게 하시오!

```
select e.ename, e.age, e.gender, e.telecom, t.price
    from emp12 e, telecom_price t
    where e.telecom = t.telecom and e.age>= 27;
```

231. 다음 결과를 다시 출력하는데 컬럼명을 한글로 사원, 관리자라고 출력되게 하시오.

(보기)

```
select 사원.ename, 관리자.ename
    from emp 사원, emp 관리자
    where 사원.mgr =관리자.empno;
```

>>>

(정답)

```
select 사원.ename as 사원, 관리자.ename as 관리자
    from emp 사원, emp 관리자
    where 사원.mgr =관리자.empno;
```

232. 사원이름, 사원월급, 관리자이름, 관리자의 월급을 출력하시오!

```
select 사원.ename as 사원, 사원. sal as 사원월급, 관리자.ename as 관리자,
                                        관리자.sal as "관리자의 월급"
                                                <<< 스페이스 주려면 더블쿼테이션 쓰기
    from emp 사원, emp 관리자
    where 사원.mgr = 관리자.empno;
```

233. 위의 결과를 다시 출력하는데 사원의 월급이 관리자의 월급보다 더 큰 사람들만 출력하시오!

```
select 사원.ename as 사원, 사원. sal as 사원월급, 관리자.ename as 관리자,
                                        관리자.sal as "관리자의 월급"
    from emp 사원, emp 관리자
```

where 사원.mgr = 관리자.empno and 사원.sal> 관리자.sal;

234. 관리자보다 먼저 입사한 사원들의 사원 이름과 사원 입사일, 관리자 이름, 관리자 입사일을 출력하시오!

```
select 사원.ename, 사원.hiredate, 관리자.ename, 관리자.hiredate
    from emp 사원, emp 관리자
    where 사원.mgr = 관리자.empno and 사원.hiredate < 관리자.hiredate;
```

235. 관리자의 이름을 출력하고 그 옆에 해당 관리자에게 속한 사원들의 이름을 가로로 출력하시오!

```
select 관리자.ename as 관리자,
            listagg(사원.ename, ', ') within group (order by 사원.ename asc) as 사원
    from emp 사원, emp 관리자
     where 사원.mgr = 관리자.empno
     group by 관리자.ename;
```

236. 직업, 이름, 월급, 순위를 출력하는데 순위가 월급이 높은 순서대로 순위를 부여하시오!

```
select job, ename, sal, dense_rank() over (order by sal desc) as 순위
    from emp;
```

237. 위의 결과를 다시 출력하는데 직업별로 각각 순위가 부여되게 하시오.

```
select job, ename, sal, dense_rank() over (partition by job
                                    order by sal desc) as 순위
    from emp;
```

238. 부서위치, 이름, 월급, 순위를 출력하는데 부서위치별로 각각 월급이 높은 순서대로 순위를 부여하시오!

```
select d.loc, e.ename, e.sal, rank() over (partition by d.loc
                                    order by e.sal desc) as 순위
    from emp e, dept d
    where e.deptno = d.deptno;
```

239. 부서위치, 부서위치별로 속한 사원들의 이름을 가로로 출력하시오!

```
select d.loc, listagg(e.ename, ', ') within group(order by e.ename asc) as 사원
    from emp e, dept d
    where e.deptno = d.deptno
    group by d.loc;
```

240. 부서위치, 부서위치별 토탈월급을 출력하는데 맨 아래에 전체 토탈월급이 출력되게 하시오!

```
select d.loc, sum(e.sal)
    from emp e, dept d
    where e.deptno = d.deptno
    group by rollup(d.loc);
```

241. 직업, 직업별 최대월급, 직업별 최소월급, 직업별 평균월급을 출력하시오!

```
select job, max(sal), min(sal), avg(sal)
    from emp
    group by job;
```

242. 부서위치, 부서위치별 최대월급, 부서위치별 최소월급, 부서위치별 인원수를 출력하시오

```
select d.loc, max(e.sal), min(e.sal), count(*)
    from emp e, dept d
    where e.deptno = d.deptno
    group by d.loc;
```

243. 이름과 월급과 부서위치를 출력하는데 월급이 2400 이상인 사원들만 출력하시오!
(on 절을 사용한 조인 문법으로 수행)

```
select e.ename, e.sal, d.loc
    from emp e join dept d
    on (e.deptno = d.deptno) <<< 조인조건
    where e.sal >= 2400; <<< 검색조건
```

❖ 설명 : on절을 사용한 조인 문법은 조인조건은 on절에 주게 되어있고 검색조건은 where절에 주게끔 구분해 놓았다. 이렇게 해야 에러가 안난다.

244. DALLAS에서 근무하는 사람들의 이름과 월급과 부서위치와 직업을 출력하는데 on절을 사용한 조건 문법으로 수행하세요~

```
select e.ename, e.sal, d.loc, e.job
    from emp e join dept d
    on(e.deptno = d.deptno)
    where d.loc = 'DALLAS';
```

245. emp테이블과 salgrade 테이블을 서로 조인해서 이름, 월급, 급여등급(grade)을 출력하는데 2등급만 출력되게 하고 on절을 사용한 조인문법으로 수행하시오!

```
select e.ename, e.sal, s.grade
    from emp e join salgrade s
    on (e.sal between s.losal and s.hisal) <<< 조인조건
    where s.grade = 2; <<< 검색조건
```

246. 직업이 SALESMAN인 사원들의 이름과 월급과 직업, 부서위치를 출력하는데 USING절을 사용한 조인문법으로 수행하시오!

```
select e.ename, e.sal, e.job, d.loc
    from emp e join dept d
    using (deptno) <<< 연결고리의 컬럼
    where e.job = 'SALESMAN'; <<< 검색조건
```

247. 다음 결과를 1999 ansi 문법을 구현하시오.
(보기)

select e.ename, d.loc
from emp e, dept d
where e.deptno = d.deptno(+)

(정답)

```
select e.ename, d.loc
    from emp e left outer join dept d
    on (e.deptno = d.deptno);
```

248. 우리반 테이블과 telecom_price와 조인을 해서 이름이 김정민 학생의 이름과 나이와 통신사와 통신요금(price)을 출력하시오!

```
select e.ename, e.age, e.telecom, t.price
    from emp12 e, telecom_price t
    where e.telecom = t.telecom and e.ename = '김정민';
```

또는

```
select e.ename, e.age, e.telecom, t.price
    from emp12 e join telecom_price t
    on (e.telecom = t.telecom)
   where e.ename = '김정민';
```

249. 나이가 28 이상인 학생들의 이름과 나이와 통신사와 통신요금(price)를 출력하시오!

```
select e.ename, e.age, e.telecom, t.price
    from emp12 e join telecom_price t
    on (e.telecom = t.telecom)
    where e.age >= 28;
```

250. 부서위치, 부서위치별 토탈월급을 가로로 출력하시오!

보기>>

NEW YORK	**DALAS**	**CHICAGO >>> 컬럼명**
8750	**10875**	**9400 >>> 데이터**

```
select sum(decode(d.loc, 'NEW YORK', e.sal)) as "NEW YORK",
            sum(decode(d.loc, 'DALLAS', e.sal)) as "DALLAS",
            sum(decode(d.loc, 'CHICAGO', e.sal)) as "CHICAGO"
    from emp e, dept d
    where e.deptno = d.deptno;
```

또는

```
select sum(decode(d.loc, 'NEW YORK', e.sal, null)) as "NEW YORK",
        sum(decode(d.loc, 'DALLAS', e.sal, null)) as "DALLAS",
        sum(decode(d.loc, 'CHICAGO', e.sal, null)) as "CHICAGO"
    from emp e, dept d
    where e.deptno = d.deptno;
```

>>>null은 안 써줘도 된다.

pivot 문으로 하는 법:

※ 설명 : 위의 결과를 보기 위해 필요한 raw data는 부서위치와 월급입니다. 그러므로 pivot문의 from 절에 부서위치와 월급 컬럼만 가져오는 쿼리문을 작성하면 됩니다.

```
select *
    from (select d.loc, e.sal
            from emp e, dept d
            where e.deptno = d.deptno)
    pivot (sum(sal) for loc in ('NEW YORK' as "NEW YORK", 'DALLAS' as DALLAS, 'CHICAGO' as CHICAGO));
```

※ 설명 : from 절 바로 안에 조인쿼리문은 결과를 보기 위한 raw data입니다. from절의 조인 쿼리문에서 d.loc 라고 작성했지만 실제로 출력되는 컬럼명은 loc이므로 pivot문에서 d.loc라고 하면 안되고 loc라고 작성해줘야 합니다.
pivot문에서는 출력되는 대로 써줘야 하기 때문

오늘의 문제 (251-467)

2020년 11월 4일 수요일 오전 10:09

251. 부서번호, 부서번호별 토탈월급을 출력하는데 맨 아래에 전체 토탈월급이 출력되게 하시오! (union all로 수행)

```
select to_char(deptno), sum(sal)
    from emp
    group by deptno
union all
select '전체토탈:' as deptno, sum(sal)
    from emp;
```

deptno는 숫자형이고 '전체토탈'은 문자형이기 때문에 둘을 맞춰주기 위해 윗 문장 deptno에 to_char를 붙여서 맞춰준다.

252. 입사한 년도(4자리). 입사한 년도별 토탈월급을 출력하시오!

```
select to_char(hiredate, 'RRRR'), sum(sal)
    from emp
    group by to_char(hiredate, 'RRRR');
```

253. 입사한 년도(4자리), 입사한 년도별 토탈월급을 출력하는데 맨 아래에 전체토탈월급이 출력되게 하시오.

```
select to_char(hiredate, 'RRRR'), sum(sal)
    from emp
    group by to_char(hiredate, 'RRRR')
union all
select '전체토탈:' as hiredate, sum(sal)
    from emp;
```

또는

```
select to_char(hiredate, 'RRRR') as hire_year, sum(sal)
    from emp
    group by to_char(hiredate, 'RRRR')
union all
select '토탈:' as hire_year, sum(sal)
    from emp;
```

※ 설명 : union all 위 아래로 쿼리문의 컬럼의 갯수가 서로 동일해야하고 컬럼명도 가급적 동일해야하고 데이터 유형도 동일해야 합니다.

254. 우리반 테이블에서 통신사, 통신사별 인원수를 출력하시오.

```
select telecom, count(*) <<< 가급적 * 쓰기
    from emp12
    group by telecom;
```

255. 아래와 같이 전체 인원수가 맨 아래에 출력되게 하시오.

<보기>

sk 11
lg 4
kt 15
전체: 30

```
select telecom, count(*)
    from emp12
    group by telecom
union all
select '전체:' as telecom, count(*)
    from emp12;
```

union all을 계속 쓰고싶으면 밑에 계속 union all 쓰고 select문장 쓰고 하면됨.

256. 문제 253번의 결과를 정렬해서 출력하시오!

(보기)
1981 22825 ---> 1980 800
1983 1100 ---> 1981 22825
1980 800 ---> 1982 4300
1982 4300 ---> 1983 1100
토탈: 29025 ---> 토탈 : 29025

```
select to_char(hiredate, 'RRRR') as hire_year, sum(sal)
    from emp
    group by to_char(hiredate, 'RRRR')
union all
select '토탈:' as hire_year, sum(sal)
    from emp
    order by hire_year asc;
```

※ 설명 : order by절은 항상 맨 아래의 쿼리문에만 사용해야 합니다.

257. 부서번호, 부서번호별 토탈월급을 출력하는데 맨 아래에 전체토탈 월급이 출력되게 하고 부서번호를 10,20,30번 순으로 정렬해서 출력되게 하시오.

```
select to_char(deptno), sum(sal)
    from emp
    group by to_char(deptno)
union
select '전체토탈:' as deptno, sum(sal)
    from emp;
```

※ 설명: union은 암시적으로 정렬 작업을 수행하기 때문에 굳이 정렬된 결과를 볼 필요가 없다면 union all을 사용하는게 더 검색 성능에 좋습니다.

258. 부서위치, 부서위치별 토탈월급을 출력하시오. (세로 출력)

```
select d.loc, sum(e.sal)
    from emp e, dept d
    where e.deptno = d.deptno
    group by d.loc;
```

259. 맨 아래에 전체 토탈월급도 출력되게 하시오!

```
select d.loc, sum(e.sal)
    from emp e, dept d
    where e.deptno = d.deptno
    group by d.loc
union all
select '전체토탈:' as loc, sum(sal)
    from emp; >>> loc 출력 안해도 되니까 굳이 조인할 필요 X
```

260. 위의 결과를 다시 출력하는데 직업을 ABCD 순서대로 정렬해서 출력되게 하시오.

```
select d.loc, sum(e.sal)
    from emp e, dept d
    where e.deptno = d.deptno
    group by d.loc
union
select '전체토탈:' as loc, sum(sal)
    from emp;
```

※ 설명 : 정렬을 할 필요가 없으면 union all을 쓰고 정렬이 필요하면 union을 사용합니다.

261. 직업, 직업별 최대월급, 직업별 최소월급, 직업별 평균월급, 직업별 인원수를 출력하시오.

```
select job, max(sal), min(sal), avg(sal), count(*)
    from emp
    group by job;
```

262. 사원 테이블에서 최대월급, 최소월급, 평균월급, 전체인원수를 출력하시오!

```
select max(sal), min(sal), avg(sal), count(*)
    from emp;
```

263. 문제 261번 결과와 문제 262번의 결과를 위아래로 연결해서 출력하시오!

(보기)

SALESMAN	**1600**	**1250**	**1400**	**4**
CLERK 1300	**800**	**1037.54**		
ANALYST	**3000**	**3000**	**3000**	**2**
MANAGER	**2975**	**2450**	**2758.3333**	**3**
PRESIDENT	**5000**	**5000**	**5000**	**1**
전체:	**5000**	**800**	**2073.2142**	**14**

```
select job, max(sal), min(sal), avg(sal), count(*)
    from emp
    group by job
union all
select '전체:' as job, max(sal), min(sal), avg(sal), count(*)
    from emp;
```

264. 위의 결과에서 숫자에 전부 천단위가 부여되게 하시오!

```
select job, to_char(max(sal), '999,999'), to_char(min(sal), '999,999'), to_char(avg(sal), '999,999') , count(*)
    from emp
    group by job
union all
select '전체:' as job, to_char(max(sal), '999,999'), to_char(min(sal), '999,999'), to_char(avg(sal), '999,999'),
count(*)
```

```
from emp;
```

265. SCOTT과 같은 월급을 받는 사원들의 이름과 월급을 출력하시오!
SCOTT은 제외하고 출력하시오!

```
select ename, sal
   from emp
   where sal = (select sal
                from emp
                where ename = 'SCOTT')
and ename != 'SCOTT';
```

서브쿼리메인쿼리

266. SMITH와 직업이 같은 사람들의 이름과 직업을 출력하는데 SMITH는 제외하고 출력하시오!

```
select ename, job
   from emp
   where job = (select job
                from emp
                where ename = 'SMITH')
             and ename != 'SMITH';
```

267. ALLEN보다 늦게 입사한 사원들의 이름과 입사일을 출력하시오!

```
select ename, hiredate
   from emp
   where hiredate > (select hiredate
                     from emp
                     where ename = 'ALLEN');
```

268. 직업이 SALESMAN인 사원의 최대월급보다 더 많은 월급을 받는 사원들의 이름과 월급을 출력하시오!

```
select ename, sal
   from emp
   where sal> (select max(sal)
                     from emp
                     where job = 'SALESMAN');
```

269. 최대월급을 받는 사원의 이름과 월급을 출력하시오.

```
select ename, sal
   from emp
   where sal= (select max(sal)
              from emp);
```

270. 전공에 통계가 포함되어져있는 학생들 중에서의 최대나이인 학생의 이름과 나이와 전공을 출력하시오!

```
select ename, age, major
   from emp12
   where age = (select max(age)
              from emp12
         where major like '%통계%')
   and major like '%통계%';
```

271. KING에게 보고하는 사원들의 이름을 출력하시오!
(KING에게 보고하는 사원들은 KING의 직속 부하 사원들 입니다.)

KING의 직속부하 사원들의 MGR번호는 KING의 EMPNO입니다.

```
select ename
   from emp
   where mgr = (select empno
                from emp
                where ename = 'KING');
```

272. 통신사가 sk인 학생들과 나이가 같은 학생들의 이름과 나이와 통신사를 출력하시오!

```
select ename, age, telecom
   from emp12
   where age in (select age
                 from emp12
                 where telecom = 'sk');
```

273. 통신사가 sk인 학생들과 나이가 같지 않은 학생들의 이름과 나이와 통신사를 출력하시오!

```
select ename, age, telecom
   from emp12
   where age not in (select age
                     from emp12
                     where telecom = 'sk');
```

274. 관리자가 아닌 사원들의 이름을 출력하시오!

(관리자인 사원이 6명이 나왔으므로 관리자가 아닌 사원들이 8명이 나와야 함)

```
select ename
   from emp
   where empno not in (select mgr
                from emp);
```

이렇게 하면 결과가 안 나옴.

>>> mgr이 null인 사람이 있어서 (KING)

※ 서브쿼리문에서 not in 연산자 사용할 시 주의할 사항!

서브쿼리에서 null값이 하나라도 리턴되면 결과가 출력되지 않는다.

275. 이제 설명을 들었으니 관리자가 아닌 사원들을 출력하시오!

(관리자가 아닌 사원들이 8명 있는데 8명이 나와줘야 합니다)

null때문에 출력이 안되었으므로 null이 리턴되지 않게 해줘야합니다.

```
select ename
   from emp
   where empno not in (select nvl(mgr,0)
                from emp);
```

또는

```
select ename
      from emp
      where empno not in (select mgr
                          from emp
                          where mgr is not null);
```

276. 부서 테이블에서 부서번호와 부서위치를 출력하는데 부서 테이블에 있는 부서번호중에 사원 테이블에도 존재하는 부서번호에 대한 것만 출력하시오.

```
select deptno, loc
   from dept d
   where exists( select *
```

```
                from emp e
                where e.deptno = d.deptno);
DEPTNO   LOC
------------------------
20          DALLAS
10          NEW YORK
30          CHICAGO
```

277. 이번에는 존재하지 않는 부서번호와 부서위치를 출력하시오.

```
select deptno, loc
   from dept d
   where not exists( select *
                     from emp e
                     where e.deptno = d.deptno);
DEPTNO   LOC
------------------------
40          BOSTON
```

278. 어제 마지막 문제(250번)에 deptno도 같이 출력하시오!

```
select e.deptno, sum(decode(d.loc, 'NEW YORK', e.sal)) as "NEW YORK",
       sum(decode(d.loc, 'DALLAS', e.sal)) as "DALLAS",
       sum(decode(d.loc, 'CHICAGO', e.sal)) as "CHICAGO"
   from emp e, dept d
   where e.deptno = d.deptno;
   group by e.deptno
```

279. 위의 결과에 null값 대신에 0으로 출력되게 하시오.

(보기)

```
DEPTNO  NEWYORK  DALLAS  CHICAGO
-------------------------------------------
30          0          0          9400
10          8750       0          0
20          0          16875      0
```

```
select e.deptno, nvl(sum(decode(d.loc, 'NEW YORK', e.sal)), 0) as "NEW YORK",
                 nvl(sum(decode(d.loc, 'DALLAS', e.sal)),0) as "DALLAS",
                 nvl(sum(decode(d.loc, 'CHICAGO', e.sal)),0) as "CHICAGO"
   from emp e, dept d
   where e.deptno = d.deptno
   group by e.deptno;
```

280. JAMES보다 더 많은 월급을 받는 사원들의 이름과 월급을 출력하시오. (11/5)

```
select ename, sal
   from emp
   where sal > (select sal
                from emp
                where ename = 'JAMES');
```

281. 직업, 직업별 토탈월급을 출력하시오.

```
select job, sum(sal)
   from emp
   group by job;
```

282. 위의 결과를 다시 출력하는데 직업이 SALESMAN의 토탈월급보다 더 큰 직업들만 출력하시오.

```
select job, sum(sal)
    from emp
    group by job
    having sum(sal) = (select sum(sal)
                       from emp
                       where job = 'SALESMAN');
```

❖ 설명 : 그룹함수로 조건을 주는 저는 having절 입니다. where절에 작성하면 에러가 발생합니다.

283. 부서번호와 부서번호별 인원수를 출력하는데 10번 부서번호의 인원수보다 더 큰것만 출력하시오.

```
select deptno, count(*)
    from emp
    group by deptno
    having count(*) > (select count(*)
                       from emp
                       where deptno = 10);
```

284. 직업, 이름, 월급, 순위를 출력하는데 순위가 직업별로 각각 월급이 높은 순서대로 순위를 부여하시오!

```
select job, ename, sal, dense_rank() over (partition by job
                                           order by sal desc) 순위
    from emp;
```

285. 위의 결과에서 순위가 1등인 사원들만 출력하시오!

```
select *
from ( select job, ename, sal, dense_rank() over (partition by job
                                                   order by sal desc) 순위
       from emp)
where 순위 = 1;
```

286. 우리반 테이블에서 통신사별로 나이가 가장 많은 학생의 이름과 나이와 통신사와 나이의 순위를 출력하시오!

```
select *
from ( select ename, age, telecom, dense_rank () over (partition by telecom
                                        order by age desc)순위
                                        from emp12)
where 순위 = 1;
```

287. price table의 전체 건수가 어떻게 되는지 확인하시오.

```
select count(*)
    from price;
```

M_NAME = 마트이름 또는 시장이름

A_NAME = 식료품 이름

A_PRICE = 식료품 가격

288. 서울시에서 가장 비싼 식료품의 이름과 가격과 파는 곳을 출력하시오!

```
select a_name, a_price, m_name
    from (select a_name, a_price, m_name, dense_rank() over (order by a_price desc) as 순위
          from price)
```

where 순위 = 1;

289. 사원이름, 월급, 사원 테이블의 최대월급, 사원테이블의 최소월급을 출력하시오.

```
select ename, sal, (select max(sal), min(sal)
                    from emp)
  from emp;
```

이렇게 하면 값이 안나온다.

❖ 중요설명 : select 절의 서브쿼리인 스칼라 서브쿼리의 특징?
스칼라 서브쿼리는 단 한개값만 리턴 할 수 있다.

```
select ename, sal, (select max(sal)
                    from emp) 최대, (select min(sal)
                                     from emp)최소
  from emp;
```

이렇게 적어주어야 한다.

290. 이름, 나이, 우리반 나이의 평균나이를 출력하시오.

```
select ename, age, (select avg(age)
                    from emp12) 평균나이
  from emp12;
```

291. 위의 결과를 소수점이하가 안나오게 반올림하시오!

```
select ename, age, (select avg(age)
                    from emp12)
  from emp12;
```

292. (난이도 상) 위의 결과에서 학생의 나이가 평균 나이보다 더 큰 학생들의 이름과 나이와 평균나이를 출력하시오.

```
select *
  from (select ename, age, (select round(avg(age))
                            from emp12) 평균나이
                            from emp12)
  where age > 평균나이;
```

**293. 사원테이블에서 이름, 월급, 사원테이블의 최대월급,
사원테이블의 최소월급,
사원테이블의 평균월급을 출력하시오.**

(튜닝전)

```
select ename, sal, (select max(sal) from emp) 최대월급,
                   (select min(sal) from emp) 최소월급,
                   (select avg(sal) from emp) 평균월급
  from emp;
```

(튜닝 후)

```
select ename, sal, max(sal) over () 최대월급,
                  min(sal) over () 최소월급,
                  avg(sal) over () 평균월급
   from emp;
```

>>> 밑에것은 emp를 한번만 돌리니까 훨씬 빠르다. 같은 값이 나오지만

294. 우리반 테이블에서 이름, 나이, 우리반 나이의 최대나이,
우리반 나이의 최소나이,
우리반 나이의 평균나이,
우리반 학생들의 인원수를 출력하시오.

```
select ename, age, max(age) over () 최대나이,
                  min(age) over () 최소나이,
                  avg(age) over () 평균나이,
                  count(*) over () 인원수
   from emp12;
```

❖ 설명 : select절의 서브쿼리인 스칼라 서브쿼리가 성능이 느리므로 위와 같이 데이터 분석 함수를 이용해서 튜닝을 하면 빠르게 대용량 데이터의 데이터를 검색할 수 있습니다.

295. 사원 테이블에 아래의 데이터를 입력하시오.

사원번호 2939
사원이름 jane
월급 4700
입사일 : 오늘날짜

```
insert into emp(empno, ename, sal, hiredate)
   values(2939, 'jane', 4700, sysdate);
```

이렇게 해주지 말고

```
insert into emp(empno, ename, sal, hiredate)
        values(2939, 'jane', 4700, to_date('20/11/05', 'RR/MM/DD'));
```

이렇게 해주는게 좋다.

❖ 중요설명 : 날짜를 입력할 때는 to_date를 사용해서 입력하세요~
실패하지 않고 확실하게 날짜 데이터를 입력하는 방법입니다.
위와같이 날짜를 입력하면 날짜는 정확하게 2020년 11월 05일로 입력이 되고 시분초는 00시 00분 00초로 입력됩니다.

296. 오늘 입사한 사원의 이름과 입사일을 출력하시오!

```
select ename, hiredate
from emp
where hiredate = sysdate;
```

>>> 이렇게 하면 조회가 안된다.

❖ 설명 : 왜 조회가 되지 않는가? sysdate는 날짜 뿐만 아니라 시분초도 출력되기 때문에 아까 입력했을 때

시분초와 지금 조회했을 때 시분초가 서로 다르기 때문에 조회되지 않는 것 입니다.

297. 2020년 11월 05일에 입사한 사원의 이름과 입사일을 출력하시오!

```
select ename, hiredate
    from emp
    where hiredate = to_date( '20/11/05','RR/MM/DD');
              여기다 함수쓰는 것은 좋지 않다.
```

❖ 설명 : where절에 컬럼명쪽에 가급적 함수를 사용하지 않고 검색하는 것이 더 빠르게 데이터를 검색할 수 있는 방법입니다.

298. 아래의 데이터를 사원 테이블에 입력하시오!

```
insert into emp(empno, ename, sal, deptno)
values(4945, 'mike    ', 3000, 20 ) ;
```

299. 이름이 mike인 사원의 이름과 월급을 출력하시오!

(튜닝 전)

```
select ename, sal
       from emp
       where rtrim(ename) = 'mike'
```

(튜닝 후)

```
select ename, sal
       from emp
       where ename like 'mike%'
```

❖ 설명 : where절에 컬럼에는 함수를 가급적 사용하지 않아야 검색 성능이 좋아집니다.

300. 아래의 데이터를 우리반 테이블에 입력하시오!

이름: 김인호

성별 : 남

이메일 : abcd1234@gmail.com

주소 : 서울시 강남구 역삼동 삼원빌딩 4층

```
insert into emp12 (ename, gender, email, address)
values('김인호', '남', 'abcd1234@gmail.com', '서울시 강남구 역삼동 삼원빌딩 4층');
```

301. KING의 월급을 9000으로 변경하시오!

```
update emp
set sal = 9000
where ename = 'KING';
```

commit; : 지금까지 변경한 모든 작업을 다 database에 영구히 저장하겠다.

rollback; : commit이후에 작업한 모든 변경사항을 취소하겠다.

302. 직업이 SALESMAN인 사원들의 커미션을 9500으로 수정하시오!

```
update emp
set comm = 9500
```

```
where job = 'SALESMAN';
```

303. 직업이 SALESMAN인 사원들을 삭제하시오.

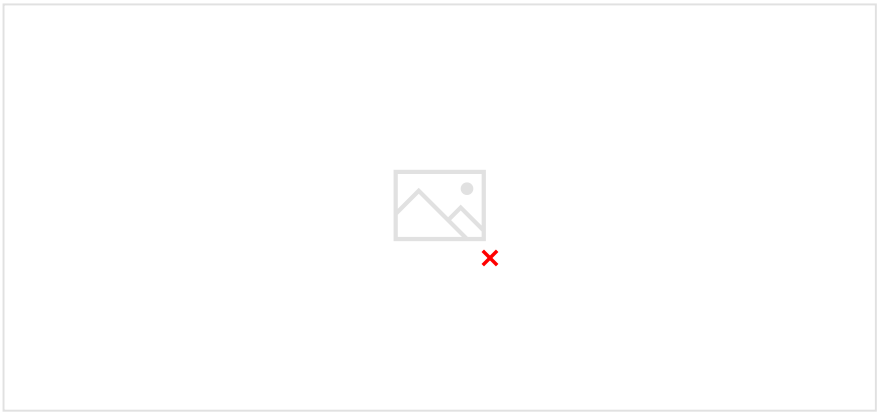
```
delete from e
mp
where job = 'SALESMAN';
```

304. 부서번호, 부서번호별 토탈월급을 출력하는데 가로로 출력하시오.

(sum + decode 이용)

```
select job, sum(decode, deptno, 10, sal)) as "10",
       sum(decode, deptno, 20, sal)) as "20",
       sum(decode, deptno, 30, sal)) as "30"
from emp
group by job;
```

305.



튜닝전 :

```
select job, sum(decode (deptno, 10, sal)) as "10",
            sum(decode(deptno, 20, sal)) as "20",
            sum(decode(deptno, 30, sal)) as "30", sum(sal) as 토탈값
    from emp
    group by job
union all
select '전체토탈: ' as job, sum(decode (deptno, 10, sal)) as "10",
                            sum(decode(deptno, 20, sal)) as "20",
                            sum(decode(deptno, 30, sal)) as "30", sum(sal)
    from emp;
```

튜닝 후:

```
select nvl(job,전체토탈), sum(decode(deptno, 10,sal)) as "10",
                          sum(decode(deptno, 20,sal)) as "20",
                          sum(decode(deptno, 30,sal)) as "30",
                          sum(sal)
       from emp
       group by grouping sets(job,());
```

306. (11/6) rollback이 마지막 commit한 시점까지만 rollback이 된다는 것을 테스트하기 위해서 commit을 지금 수행하고 사원테이블의 월급을 모두 다 0으로 변경하시오.

```
update emp
      set sal = 0;
```

307. 우리반 테이블에 telecom_price 컬럼을 추가하고 해당 텔레콤의 요금으로 값을 갱신하시오.

```
alter table emp12
   add telecom_price number(10);

merge into emp12 e
   using telecom_price t
   on (e.telecom = t.telecom)
   when matched then
   update set e.telecom_price = t.price;
```

308. 부서위치, 부서위치별 토탈월급을 출력하시오.

```
select loc, sum(sal)
   from emp
   group by loc;
```

(emp테이블에 loc 추가했으니)

309. 부서번호, 부서번호별 토탈월급을 출력하시오!

```
select deptno, sum(sal)
   from emp
   group by deptno;
```

310. 사원 테이블과 급여등급(salgrade) 테이블을 조인해서 이름과 월급과 등급(grade)을 출력하시오!

```
select e.ename, e.sal, s.grade
   from emp e, salgrade s
   where e.sal between s.losal and s.hisal;
```

311. 사원 테이블에 급여등급 (grade)컬럼을 추가하시오!

```
alter table emp
      add grade number(10);
```

312. 사원테이블에 추가한 grade컬럼에 해당 사원의 급여등급으로 값을 갱신하시오!

```
merge into emp e
   using salgrade s
   on(e.sal between s.losal and s.hisal)
   when matched then
   update set e.grade = s.grade;
```

313. 부서명 (dname) 컬럼을 emp 테이블에 추가하고 해당사원의 부서명으로 emp 테이블의 dname을 갱신하시오!

```
alter table emp
add dname varchar2(10);

merge into emp e
      using dept d
      on (e.deptno = d.deptno)
      when matched then
      update set e.dname = d.dname;
```

314. dept테이블을 백업하시오.

```
create table dept_backup
as
select *
   from dept;
```

315. dept 테이블을 truncate 하시오

```
truncate table dept;
```

316. dept 테이블을 dept_backup 테이블을 이용해서 복구하시오!

```
insert into dept
   select *
   from dept_backup;
```

317. ALLEN보다 더 늦게 입사한 사원들의 커미션을 9000으로 수정하시오!

```
update emp
   set comm = 9000
   where hiredate > (select hiredate
                from emp
                where ename = 'ALLEN');
```

318. SMITH와 같은 직업을 갖는 사원들의 월급을 9800으로 변경하시오. 그런데 SMITH는 제외하시오.

```
update emp
   set sal = 9800
   where job = ( select job
                     from emp
                     where ename = 'SMITH')
         and ename != 'SMITH';
```

319. ALLEN의 월급을 KING의 월급으로 변경하시오!

```
update emp
   set sal = (select sal
               from emp
               where ename = 'KING')
   where ename = 'ALLEN';
```

320. JONES보다 월급이 많은 사원들의 직업을 MARTIN의 직업으로 변경하시오!

```
update emp
   set job = (select job
            from emp
            where ename = 'MARTIN')
   where sal > (select sal
                from emp
                where ename = 'JONES');
```

321. ALLEN보다 늦게 입사한 사원들을 삭제하시오!

```
delete from emp
   where hiredate > ( select hiredate
                    from emp
                    where ename = 'ALLEN');
```

322. JONES와 같은 부서번호에서 일하는 사원들을 삭제하시오! 그런데 JONES는 제외시키시오!

```
delete from emp
```

```
where deptno = (select deptno
                from emp
                where ename = 'JONES')
      and ename != 'JONES';
```

323. 부서번호, 부서번호별 인원수를 출력하시오!

```
select deptno, count(*) 인원수
  from emp
  group by deptno;
```

324. 부서테이블에 cnt라는 컬럼을 추가하시오!

```
alter table dept
    add cnt number(10);
```

325. 지금 dept테이블에 추가한 cnt컬럼에 해당 부서번호에 인원수로 값을 갱신하시오!

```
merge into dept d
  using (select deptno, count(*) 인원수
         from emp
         group by deptno)e
  on (e.deptno = d.deptno)
  when matched then
  update set d.cnt = e.인원수;
```

※ 설명 : using 절에 사용한 서브쿼리문의 결과가 마치 테이블처럼 이 merge문에서 사용되고 있다.

326. (알고리즘 문제 2)

1부터 10사이의 숫자를 출력하는데 짝수만 출력하시오!

```
2
4          힌트: select level
6                from dual
8                where ?
10               connect by level <=10;
```

```
select level
  from dual
  where level not in (1,3,5,7,9)
  connect by level  <= 10;
```

(알고리즘 1번문제는 89번 개념 부분에 있습니다)

327. 밑수가 자연상수 (e)이고 진수가 10인 로그값을 출력하시오!

```
select ln(10) from dual;
```

$\log_e 10$

328. 자연상수(e)의 10승은 얼마인가?

e^{10}

```
select exp(10) from dual;
```

329. 1부터 10까지의 곱을 SQL로 출력하시오!

(log1+ log2+ log3+ log4 + ...+ log10)

e
= 1*2*3*4*5 ..... *10

```
select exp(sum(ln(level)))
    from dual
    connect by level <=10;
```

330. (11/9) emp 테이블의 서열을 SQL로 시각화 하시오!

```
select rpad(' ', level*2)|| ename as emplotee, level from emp
       start with ename = 'KING'
       connect by prior empno = mgr;
```

❖ 설명 : rpad(' ', level*2) 은 공백(' ')을 level*2의 숫자만큼 채워넣겠다.

```
KING          1
  JONES       2
    SCOTT     3
      ADAMS   4
    FORD      3
      SMITH   4
  BLAKE       2
    ALLEN     3
    WARD      3
    MARTIN    3
    TURNER    3
    JAMES     3
  CLARK       2
    MILLER    3
```

SQL은 시각화가 안되기 때문에 이렇게 함으로써 시각화를 준다.

331. 위의 결과에서 BLAKE는 제외하고 출력하시오!

```
select rpad(' ', level*2)|| ename as emplotee, level
    from emp
    where ename != 'BLAKE'
    start with ename = 'KING'
    connect by prior empno = mgr;
```

332. 위의 결과에서 BLAKE뿐만 아니라 BLAKE의 팀원들까지 전부 제외하시오!

```
select rpad(' ', level*2)|| ename as emplotee, level
    from emp
    start with ename = 'KING'
    connect by prior empno = mgr and ename != 'BLAKE'
```

❖ 하위노드의 오픈 데이터를 출력되지 않게 하려면 where절이 아니라 connect by절에 조건을 주면 됩니다.

333. 다시 BLAKE와 BLAKE 팀원들을 포함시킨 서열을 출력하는 sql을 아래와 같이 실행하는데 월급이 높은 순서대로 출력하시오

```
>>>
select rpad(' ', level*2)|| ename as employee, level from emp
       start with ename = 'KING'
       connect by prior empno = mgr;
```

정답:

```
select rpad(' ', level*2)|| ename as emplotee, sal, level from emp
    start with ename = 'KING'
    connect by prior empno = mgr
    order by sal desc;
```

❖ 설명 : 위의 결과는 월급이 높은 순서대로 정렬이 되면서 서열로 정렬된 결과가 사라져버렸다.

334. 위의 결과를 다시 서열로 정렬된 결과를 유지하면서 월급이 높은 순서대로 정렬돼서 출력되게 하시오!

```
select rpad(' ', level*2)|| ename as employee, sal, level from emp
    start with ename = 'KING'
    connect by prior empno = mgr
    order siblings by sal desc;
```

❖ 설명 : 결과를 보면 같은 서열 내에서 월급이 높은 순서대로 정렬이 되고 있습니다. 계층형 질의문을 사용할 때 order by절을 쓸 때는 siblings라는 키워드를 짝꿍으로 사용해야 합니다.

335. 이름과 입사일과 서열 순위를 출력하는데 서열 순위의 정렬상태를 유지하면서 먼저 입사한 사원순으로 정렬이 되어서 출력되게 하시오!

```
select  rpad(' ', level*2)|| ename, hiredate, level
    from emp
    start with ename = 'KING'
    connect by prior empno = mgr
    order siblings by hiredate asc;
```

336. 아래 결과에서 앞에 /를 아래와 같이 잘라버리시오

```
select ename, sys_connect_by_path(ename, '/') as path
        from emp
        start with ename = 'KING'
        connect by prior empno = mgr;
```

결과 :

```
KING   KING
JONES  KING/JONES
SCOTT  KING/JONES/SCOTT
ADAMS   KING/JONES/SCOTT/ADAMS
FORD   KING/JONES/FORD
SMITH  KING/JONES/FORD/SMITH
BLAKE  KING/BLAKE
ALLEN  KING/BLAKE/ALLEN
WARD  KING/BLAKE/WARD
MARTIN   KING/BLAKE/MARTIN
TURNER   KING/BLAKE/TURNER
JAMES  KING/BLAKE/JAMES
CLARK  KING/CLARK
MILLER   KING/CLARK/MILLER
>>>
```

답 :

```
select ename, ltrim(sys_connect_by_path(ename, '/'), '/') as path
        from emp
        start with ename = 'KING'
        connect by prior empno = mgr;
```

337. 이름, 서열, 월급, 급여등급(grade)을 출력하시오!

emp와 salgrade를 조인하시오!

```
select rpad(' ', level*2)||e.ename, level, e.sal, s.grade
    from emp e, salgrade s
    where e.sal between s.losal and s.hisal
    start with e.ename = 'KING'
    connect by prior empno = mgr;
```

338. 위의 결과를 다시 출력하는데 서열의 정렬을 유지하면서 급여등급이 높은 사원부터 출력되게 하시오!

```
select rpad(' ', level*2)||e.ename, level, e.sal, s.grade
    from emp e, salgrade s
    where e.sal between s.losal and s.hisal
    start with e.ename = 'KING'
    connect by prior empno = mgr
    order siblings by s.grade desc;
```

339. DALLS에서 근무하는 사원들의 이름, 서열, 부서위치를 출력하시오! 서열은 전체 사원을 기준으로 서열 순위를 부여하시오!)

```
select rpad(' ', level*2)|| e.ename, level, d.loc
    from emp e, dept d
    where e.deptno = d.deptno and d.loc = 'DALLAS'
    start with ename = 'KING'
    connect by prior empno = mgr;
```

340. (SQL 알고리즘 4번) 계층형 질의문을 이용해서 구구단 2단을 출력하시오!

```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
.
.
.
2 x 9 = 18
```

```
select  '2x'|| level||'='||2*level
    from dual
    connect by level <= 9;
```

341. (SQL 알고리즘 5번) 계층형 질의문을 이용해서 삼각형을 출력하시오!

```
select lpad ('✿', level, '✿')
        from dual
        connect by level <= 10;
```

❖ 설명 : select문에서 ✿을 출력하는데 level수만큼 자리를 잡고 ✿을 하나 출력하고 나머지 자리에 ✿를 채워넣어라!

```
lpad ('✿', 1,'✿')
lpad ('✿' , 2, '✿')
.
.
.
```

이런식으로

342. (SQL 알고리즘 문제 6번) 아래와 같이 결과를 출력하시오!

(힌트 : union이나 union all을 사용하시오)

✿
✿✿
✿✿✿
✿✿✿✿
✿✿✿✿✿
✿✿✿✿
✿✿✿
✿✿
✿

```
select lpad ('✿', level, '✿')
   from dual
   connect by level <= 5
union all
select lpad('✿',5-level,'✿')
from dual
connect by level <=4;
```

또는

```
select case when level < = 5 then lpad('$',level,'$')
                    else  lpad('$',10-level,'$') end
     from dual
     connect by level < = 9;
```

343. emp500 테이블(093 설명 참조)에 아래의 데이터를 입력하시오!

1111 scott 3000
2222 smith 2900

```
Insert into emp500 (empno, ename, sal) <<< 전체를 넣을거라 안넣어도됨
    values('1111', 'scott', 3000);

Insert into emp500 (empno, ename, sal)
    values('2222', 'smith', 2900);

commit;
```

344. 아래의 테이블을 생성하는 이름을 emp501로 해서 생성하시오.

empno
ename
sal
hiredate
deptno

```
create table emp501 <---테이블명은 문자로 시작해야함
(empno number(10),
ename varchar2(20),
sal number(10),
hiredate date, <---- 9999년 12월 31일까지의 날짜 데이터 입력 가능
deptno number(10));
```

345. 아래의 emp501 테이블에 데이터를 2건 입력하시오!

```
insert into emp501
    values (7963, 'BLAKE', 2850, to_date('81/05/01', 'RR/MM/DD'), 30);
```

```
insert into emp501
    values (7839, 'KING', 5000, to_date('81/11/17', 'RR/MM/DD'), 10);
```

346. 겨울왕국 대본을 입력하기 위한 테이블을 winter_kingdom이라는 이름으로 생성하시오!

```
create table winter_kingdom
(win_text long);
```

347. 영화 겨울왕국 대본에는 elsa가 많이 나오는가? anna가 많이 나오는가?

- 정규식 함수인 regexp_count를 이용하면 쉽게 구현할 수 있습니다.

```
select regexp_count(lower(win_text), 'elsa') as cnt
      from winter_kingdom;
```

❖ 설명 : win_text를 소문자로 다 변경하고 win_text 컬럼의 데이터 중에 elsa라는 단어가 몇번 나오는지 count한다.

---> winter_kingdom테이블을 long으로 만들어서 답이 안나옴. varchar로 만들어야 함.

```
select regexp_count(lower(win_text), 'elsa') as cnt
      from winter_kingdom;
```

❖ 설명 : win_text를 전부 소문자로 변경하고 regexp_count를 이용해서 스크립트 한행 한행을 다 살펴봐서 elsa라는 단어가 포함되어져 있으면 카운트 된다.

```
select sum(regexp_count(lower(win_text), 'elsa')) as cnt
from winter_kingdom;
>>> 329
```

```
select sum(regexp_count(lower(win_text), 'elsa')) as cnt
from winter_kingdom;
>>> 685
```

❖ 설명 : 카운트된 숫자들을 sum함수를 이용해서 다 더한다.

348. 겨울왕국 테이블을 drop하고 다시 생성하는데 데이터 타입을 long으로 하지말고 varchar2(4000)으로 하시오!

```
drop table winter_kingdom;

create table winter_kingdom
(win_text varchar2(4000));
```

349. 긍정단어를 저장하기 위한 테이블을 positive라는 이름으로 아래와 같이 생성하시오!

```
create table positive
(p_text varchar2(2000));
```

350. 부정단어를 저장하기 위한 테이블을 negative라는 이름으로 아래와 같이 생성하시오!

```
create table negative
(n_text varchar2(2000));
```

351. 긍정단어는 positive 테이블에 입력하고 부정단어는 negative테이블에 입력하시오!

긍정, 부정 단어 카페에서 목록 다운 후 데이터 임포트 할 때:
헤더 체크 해제, 구분자 : 탭 , 왼쪽/오른쪽 둘러싸기 : 없음으로 설정

```
select count(*)
from positive
>>> 2007

select count(*)
from negative
>>> 4783
```

352. 겨울왕국 대본에는 긍정단어가 많은가 부정단어가 많은가?

```
select count(win_text)
      from winter_kingdom w
      where lower(win_text) in (select lower(p_text)
                                from positive);
```

>>> 위의 SQL은 잘못된 SQL입니다.
문제를 풀기 위해서는?

1. from절의 서브쿼리
2. 조인
3. regexp_substr
   을 알아야 합니다.

```
select count(*) 긍정단어
      from (select regexp_substr(lower(win_text), '[^ ]+',1,a) word
                  from winter_kingdom, (select level a
                                        from dual
                                        connect by level <= 15))
      where word in (select lower(p_text)
                           from positive);
```

353. 전국 데이터 등록금 현황 데이터를 저장할 테이블을 먼저 아래와 같이 생성하시오.
>>>카페에 가서 data 게시판에 32번글 전국 대학 insert문 삽입

354. 우리나라에서 대학등록금이 가장 비싼 학교는 어디인가?

```
select university
   from univ
   where tuition = (select max(tuition)
                   from univ);
```

355. 데이터 게시판 34번에 범죄 발생 지역 데이터를 내려받아 테이블을 생성하고 데이터를 입력하시오.

crime_type: 범죄유형
c_loc : 범죄 장소
cnt : 범죄건수

356. 범죄유형을 출력하는데 중복제거해서 출력하시오!

```
select distinct(crime_type)
```

```
from crime_loc;
```

357. 살인이 가장 많이 일어나는 장소가 어디입니까?

```
select *
    from crime_loc
    where crime_type = '살인'
    order by cnt desc
    fetch first 1 rows only;
```

358. 절도가 가장 많이 일어나는 장소가 어디인지 1위부터 3위까지 출력하시오!

```
select *
    from crime_loc
    where crime_type = '절도'
    order by cnt desc
    fetch first 3 rows only;
```

359. 데이터 게시판에서 범죄원인 데이터를 가져와서 오라클에 테이블로 구성하고 데이터를 입력하시오!

>>> 카페에서 데이터 복사

360. 살인을 일으키는 가장 큰 원인은 무엇인가?

데이터 ---> 컬럼 : pivot

컬럼 ---> 데이터 : unpivot

아래의 SQL은 as 다음에 나오는 쿼리문의 결과를 crime_cause2로 생성하겠다는 뜻 입니다.

```
create  table  crime_cause2
as
select *
from crime_cause
unpivot ( cnt for term in (생계형, 유흥, 도막, 허영심, 복수, 해고, 징벌, 가정불화, 호기심, 유혹, 사고, 불만, 부주의, 기타));
```

```
select *
      from crime_cause2
      where crime_type = '살인'
      order by cnt desc
      fetch first 1 rows only;
```

361. 동전을 던져서 앞면이 나오는지 뒷면이 나오는지 확인하시오!

(앞면은 숫자 0, 뒷면은 숫자1)

```
select dbms_random.value( 0, 1)
      from dual;
```

❖ 설명 : 0~1사이의 실수를 랜덤으로 출력하는데 가우시안 분포를 따르는 데이터로 출력이 된다.

```
select round( dbms_random.value( 0, 1))
    from dual;
```

이렇게 하면 결과를 도출할 때 마다 0 또는 1의 값으로 값이 달라진다.

362. 동전을 10번 던지시오.

```
select round( dbms_random.value( 0, 1))
       from dual
       connect by level <= 10;
```

설명 : connect by level <= 10을 썼으므로 select round(dbms_random.value(0, 1))가 10번 실행되었음.

363. (SQL 알고리즘 7번문제) 동전을 10만번 던졌을 때 동전이 앞면이 나올 확률은 얼마인가? (마지막문제)

나의 답:

```
select sum(round(dbms_random.value( 0, 1)))/100000
    from (select level
            from dual
            connect by level <=100000);
```

문제에 가까운 답:

```
select count(*) / 100000 as "동전이 앞면이 나올 확률"
   from ( select round( dbms_random.value(0, 1) ) as 동전
               from dual
               connect by level <= 100000)
   where 동전 = 0;
```

```
undefine 던진횟수
select count(*) / &&던진횟수 as "동전이 앞면이 나올 확률"
   from ( select round( dbms_random.value(0, 1) ) as 동전
               from dual
               connect by level <= &던진횟수)
   where 동전 = 0;
```

치환함수를 앞쪽엔 &&으로, 뒷쪽엔 &로 해줌으로써 위의 던진횟수와 아래 던진횟수가 같은값이 된다. 그런데 이렇게 하면 다시 실행해 줄 때 다시 안물어본다.

설명 : 던진 횟수를 SQL내에서 두번 입력해야 하는데 한번만 입력하게 하려면 엔퍼센트 (&)를 앞에 두개 쓰고 뒤에 한 개 쓰면 됩니다.
그런데 SQL내에서 치환변수시 사용하는 엔퍼센트(&)를 두개를 한번이라도 사용하게 되면 두번째 실행할때는 안물어보고 바로 전에 입력된 값으로 실행해 버립니다. 그래서 치환변수 내의 내용을 비우려면 undefine 치환변수 이름을 해줘야 합니다.

364. 아래의 SQL에서 여기에 적힌 100000(초록색 형광펜 칠해진 곳)을 하드코딩하지 않고 인라인뷰에서 시행한 동전 던진 전체 횟수가 들어가게 하려면 어떻게 해야할까?

```
select count(*) / 100000 as "동전이 앞면이 나올 확률"
   from ( select round( dbms_random.value(0, 1) ) as 동전
               from dual
               connect by level <= 100000)
   where 동전 = 0;
```

```
select (1- sum(동전)/count(*)) as "동전이 앞면이 나올 확률"
   from ( select round( dbms_random.value(0, 1) ) as 동전
```

```
            from dual
            connect by level <= 100000)
```

365. (SQL 알고리즘 문제 8번) 구구단 전체를 출력하시오!

```
2 x 1 = 2
2 x 2 = 4
.
.
.
9 x 9 = 81
```

```
select a.num1||'x'||b.num2||'='||a.num1*b.num2
   from (select level num1
         from dual
         where level !=1
         connect by level <=9)a,
         (select level num2
            from dual
            connect by level <=9) b;
```

366. 구구단 문제를 풀기 위한 from절의 서브쿼리문의 결과를 임시테이블로 각각 생성하시오.

```
select a.num1, b.num2
      from (select level as num1
                  from dual
                  connect by level <=9) a,
           (select level as num2
                  from dual
                  connect by level <=9) b;

create global temporary table num1_9
 (num1 number(10))
on commit preserve rows;

insert into num1_9
select level
      from dual
      connect by level <=9;
=========================
create global temporary table num2_9
 (num2 number(10))
on commit preserve rows;
======================
insert into num2_9
select level
      from dual
      connect by level <=9;

select * from num1_9;
================
select a.num1, b.num2
      from num1_9 a, num2_9 b
```

367. (SQL 알고리즘 9번) 주사위를 10만번 던져서 주사위의 눈이 6이 나올 확률은 어떻게 되는가?

```
select count(*) / &주사위던진횟수 as "주사위의 눈이 6이 나올 확률"
        from ( select round(dbms_random.value(0.5, 6.5)) as 주사위
```

```
                from dual
                connect by level <= &주사위던진횟수)
  where 주사위 = 6;
```

반올림 때문에 범위를 (1,6) 으로 잡으면 1과 6이 나올 확률이 낮아짐 따라서 범위를 (0.5, 6.5)로 잡아줘야 함.

368. (점심시간문제) (SQL알고리즘 10번문제) 두개의 주사위를 던져서 두개의 주사위의 눈의 합이 10이되는 확률을 구하시오!

```
select count(*)/100000
        from ( select round( dbms_random.value(0.5, 6.5) ) as 주사위1,
                round( dbms_random.value(0.5,6.5)) as 주사위2
                  from dual
                  connect by level <=100000)
        where 주사위1+주사위2=10;
```

(더 빠르게 결과를 얻는 방법)

```
select count(*)/100000
        from ( select round( dbms_random.value(0.5, 6.5) ) as 주사위1,
                round( dbms_random.value(0.5,6.5)) as 주사위2
                  from dual
                  connect by level <=100000)
        where 주사위1+주사위2=10;
```

369. DALLAS에서 근무하는 사원들의 이름과 부서위치를 출력하시오! (조인)

```
select e.ename, d.loc
   from emp e, dept d
   where e.deptno = d.deptno and d.loc = 'DALLAS';
```

370. 위의 결과 데이터를 담는 테이블 emp710을 생성하시오.

```
create table emp710
as
   select e.ename, d.loc
   from emp e, dept d
   where e.deptno = d.deptno and d.loc = 'DALLAS';
```

371. 숫자 1번부터 10번까지의 숫자를 담는 테이블을 emp705로 생성하시오

```
create table emp705
as
   select level as num1
      from dual
      connect by level <=10;
```

※ 설명 : 위의 level은 SQL문과 같은 예약어여서 예약어를 테이블의 컬럼명으로 생성할 수 없다.

372. emp705의 숫자 데이터중에 임의로 아무거나 하나를 지우시오!

```
delete from emp705
   where num1 = 1;
```

373. 1부터 10사이의 숫자중 하나가 없다. 그 수는?

카카오 면접질문의 의도 : loop문 쓰지말고 알아내시오

힌트 : 1~10 사이의 숫자의 합을 구하는 쿼리문을 from절의 서브쿼리로 만들고 작성

힌트 : 수학자 가우스가 초등학교때 1부터 10까지의 숫자의 합을 구하라

374. 직업, 직업별 토탈월급을 출력하시오 (세로 출력)

```
select job, sum(sal)
      from emp
      group by job;
```

375. 위의 결과를 출력하는 view를 생성하시오! (view이름은 emp403)

```
create view emp403
as
select job, sum(sal) 토탈
   from emp
   group by job;
```

※ 설명 : view 생성할 때 그룹함수를 사용하게 되면 컬럼별칭을 줘야합니다.

376. 부서번호, 부서번호별 평균월급을 출력하는 view를 deptno_avg라는 이름으로 생성하시오!

```
create view deptno_avg
as
select deptno, avg(sal) as 부서평균
   from emp
   group by deptno;
```

377. emp와 지금 만든 deptnno_avg view와 조인해서 이름, 월급, 부서번호, 부서평균을 출력하시오!

emp ---------- deptno_avg

```
select e.ename, e.sal, e.deptno, d.부서평균
   from emp e, deptno_avg d
   where e.deptno = d.deptno;
```

378. 이름, 월급, 부서번호, 부서평균을 출력하는데 월급이 부서평균보다 더 큰 사원들만 출력하시오!

```
select e.ename, e.sal, e.deptno, d.부서평균
   from emp e, deptno_avg d
   where e.deptno = d.deptno and d.부서평균 < e.sal;
```

379. emp테이블에서 퇴사할 것 같은 사원들을 예측하기 위해 자기의 월급이 자기가 속한 직업의 평균월급보다 더 작은 사원들의 이름과 월급과 직업과 직업평균을 출력하시오!
(직업평균보다 더 월급을 적게 받는 사원)

```
create view job_avg
as
select job, avg(sal) as 직업평균
   from emp
   group by job;

select e.ename, e.sal, e.job, j.직업평균
   from emp e, job_avg j
```

```
    where e.job = j.job and e.sal < j.직업평균;
```

380. 직업, 이름, 월급, 순위를 출력하는데 순위가 직업별로 각각 월급이 높은 순서대로 순위를 부여하시오!

```
select job, ename, sal, rank () over (partition by job
                                      order by sal desc)
    from emp;
```

381. 위의 쿼리의 결과를 view로 만들고 view를 쿼리해서 순위가 1등인 사원들만 출력하시오!

```
create view rank_salj
    as
select job, ename, sal, rank () over (partition by job
                                      order by sal desc) as 순위 << 반드시 컬럼별칭이 있어야 함
    from emp;

select *
    from rank_salj
    where 순위 = 1;
```

382. 이름과 부서위치를 출력하는 view를 ename_loc라는 이름으로 생성하시오!

```
create view ename_loc
as
select e.ename, d.loc
    from emp e, dept d
    where e.deptno = d.deptno;  (복합 view)

select *
    from ename_loc;
```

383. ename_loc의 데이터 중에 SCOTT의 부서위치를 워싱턴으로 바꾸시오.

```
update ename_loc
    set loc = 'washington'
    where ename = 'SCOTT';
```

※ 설명 : 변경을 할 수 없습니다. SCOTT의 부서위치가 DALLAS에서 WS로 변경이 된다면 실제로 DEPT 테이블의 부서위치가 DALLAS에서 WS으로 변경되는거라서 SCOTT이 아닌 다른 사원들도 DALLAS에서 WS으로 변경되어지므로 변경이 안되게 오라클에서 막은것 입니다.

문제 375번에서 만든 emp403 view를 쿼리하고 결과를 보시오.

```
create view emp403
as
select job, sum(sal) 토탈
    from emp
    group by job;

select *
      from emp403;
---------------------------------
update emp403
      set 토탈 = 2000
      where job = 'SALESMAN';
```

※ 설명 : 위와 같이 복합뷰는 데이터를 갱신할 수 없는데 만약 토탈월급이 갱신된다면 실제값도 갱신해

줘야 하기 때문에 갱신할 수도 없고 해서도 안된다.

384. 내가 그동안 만든 view들이 뭐가 있는지 확인하시오

```
select view_name, text <-- 만든 view에대한 select문장이 나온다.
    from user_views;
```

※ 설명 : 위와 같이 조회하면 view를 만들었을때 사용한 테이블명도 볼 수 있습니다. 외부에서 온 데이터 분석가들은 위의 쿼리를 조회할 수 있는 권한을 제한하는 경우가 많습니다.

385. 사원 테이블의 월급에 인덱스를 거시오!

```
create index emp_sal
    on emp(sal);
```

386. emp테이블의 sal의 인덱스인 emp_sal 구조를 확인하시오.

```
select sal, rowid
    from emp
    where sal >= 0;
```

※ 설명 : index에서 데이터를 읽어왔기 때문에 order by절을 사용 안했는데 월급이 정렬이 되어서 출력되었습니다. 검색을 뭘 하겠다고 조건을 주어야 목차를 활용합니다.

387. (SQL알고리즘 12번) 두개의 주사위를 동시에 던져서 주사위의 눈의 합이 짝수가 되는 확률이 어떻게 되는가?

```
select 2*count(*)/&던진횟수
                from ( select round(dbms_random.value(0.5, 6.5)) as 주사위1,
            round(dbms_random.value(0.5, 6.5)) as 주사위2
                    from dual
                    connect by level <= &던진횟수)
        where 주사위1 in (1,3,5) and 주사위2 in (2,4,6);
```

또는

```
select count(*)/100000
    from ( select round(dbms_random.value(0.5, 6.5)) as 주사위1, round(dbms_random.value(0.5, 6.5))
as 주사위2
                        from dual
            connect by level <= 100000)
            where mod(주사위1+주사위2,2) = 0;
```

확률의 고전적 정의?

$$\text{확률} = \frac{\text{사건 A에 속하는 원소수}}{\text{표본공간의 전체 원소수}}$$

388. (SQL 알고리즘 13번) 주사위 하나와 동전 한개를 동시에 던져서 주사위의 눈은 5가 나오고 동전은 앞면

이 나올 확률은 어떻게 되는가?

동전 앞면 : 0
동전 뒷면 : 1

```
select count(*)/100000
   from(select round(dbms_random.value(0.5, 6.5)) as 주사위, round(dbms_random.value(0, 1)) 동전
            from dual
                     connect by level <= 100000)
   where 주사위 = 5 and 동전 =0;
```

389. 사원 테이블에 월급의 인덱스를 생성하고 사원 테이블을 검색하는데 월급이 3000인 사원들의 이름과 월급을 검색하시오.

```
create index emp_sal
   on emp(sal);

select ename, sal
   from emp
   where sal > 3000;
```

390. 위의 SQL의 실행계획을 확인하시오!

--> SQL을 실행하기 위해서 계획을 세우고 SQL을 실행한다.
F10 을 눌러서 실행계획을 확인하거나 아니면 아래워 같이

```
explain for
select ename, sal
   from emp
   where sal > 3000;
```

391. 위의 SQL이 emp_sal의 인덱스를 통해서 데이터를 검색할 수 있도록 힌트를 주시오!

```
explain plan for
select */+ index(emp emp_sal) */ ename, sal
   from emp
   where sal > 3000;
select * from table(dbms_xplan.display);
```

392. 입사일에 인덱스를 생성하고 81년 11월 17일에 입사한 사원의 이름과 입사일을 조회하시오!

```
create index emp_hiredate
   on emp(hiredate);

select ename, hiredate
   from emp
   where hiredate = to_date('81/11/17', 'RR/MM/DD');
```

393. 위의 SQL의 실행계획을 확인하고 혹시 full table scan을 했으면 index scan을 할 수 있도록 튜닝하시오!

```
explain plan for
      select /*+ index(emp_hiredate) */ename, hiredate
         from emp
         where hiredate = to_date('81/11/17', 'RR/MM/DD');
```

```
select *
    from table(dbms_xplan.display);
```

394. 이름과 월급ㄹ 출력하는데 월급이 높은 사원부터 출력하시오

(order by절 쓰지 말고 인덱스를 이용하세요)

```
select /*+ index_desc(emp emp_sal) */ename, sal
        from emp
        where sal >= 0
```

❖ 설명 : where 절에 인덱스 컬럼의 조건을 주는 것은 필수입니다.
where절 없이 힌트만 사용해서는 안됩니다.

395. 아래의 SQL을 튜닝하시오.

튜닝전 :

```
select ename, hiredate
from emp
order by hiredate desc;

select /*+ index_desc(emp emp_hiredate) */ ename, sal
from emp
where hiredate < to_date('99/12/31', 'RR/MM/DD');
```

396. 사원 테이블에 직업에 인덱스를 생성하시오!

```
create index emp_job
    on emp(job);
```

397. 아래의 SQL을 튜닝하시오!

튜닝전 : select, ename, job

```
from emp
where substr(job, 1, 5) = 'SALES';
```

튜닝 후 :

```
select ename, job
    from emp
    where job like 'SALES%';
```

❖ 설명 : 왼쪽(좌변)에 있는 인덱스 컬럼을 가공하면 FULL TABLE SCAN하게 되므로 좌변을 가공하면 안된다.

398. 아래의 SQL을 튜닝하시오!

튜닝전 : select ename, hiredate

```
from emp
where to_char(hiredate, 'RRRR') = 1981
```

튜닝 후 : select ename, hiredate

```
from emp
where hiredate between to_date('1981/01/01', 'RRRR/MM/DD') and to_date('1981/12/31',
'RRRR/MM/DD');
```

399. 직업이 SALESMAN인 사원들의 이름과 월급과 직업을 출력하는데 월급이 높은 사원부터 출력하시오!

```
select /*+ index_desc(emp emp_sal) */ ename, sal
   from emp
   where job = 'SALESMAN';
```

400. 나머지 인덱스들도 전부 drop 하시오 !

```
drop  index  EMP_HIREDATE;
drop  index  EMP_JOB;
drop  index  EMP_SAL;
```

401. 백업받은 emp_backup_20201111의 데이터를 emp테이블에 입력하시오!

```
insert into emp
   select *
   from emp_backup_20201111;
```

402. 아래와 같이 salgrade 테이블을 전부 delete하고 commit한 후에 복구하시오!

```
delete from salgrade;
commit;

create table salgrade_backup_20201111
as
select *
   from salgrade as of timestamp( systimestamp - interval '10'minute);

insert into salgrade
   select *
   from salgrade_backup_20201111;
```

403. (SQL 알고리즘 14번) 스마일게이트, 엑셀 입사 알고리즘 문제

factorial을 SQL로 구연하시오!

아래와 같이 치환변수를 이용해서 숫자를 물어보게 하시오!

```
select ename, sal
from emp
where empno =&숫자


select exp(sum(ln(level)))
      from emp
      where level <= &숫자;
```

404. emp508 테이블의 ename에 걸린 unique 제약을 삭제하시오!

```
alter table emp508
   drop constraint emp508_ename_un;
```

405. 우리반 테이블의 이름에 unique 제약을 거세요.

```
alter table emp12
   add constraint emp12_ename_un unique(ename);
```

406. 사원 테이블의 월급에 not null 제약을 거시오!

```
alter table emp
      modify ename constraint emp_sal_nn not null;
```

407. 사원테이블의 월급에 걸린 not null 제약을 삭제하시오!

```
alter table emp
   drop constraint emp_sal_nn;
```

408. 사원 테이블의 월급에 월급이 0~9500 사이의 데이터만 입력되게끔 CHECK제약을 거시오!

```
alter table emp
   add constraint emp_sal_ck check( sal between 0 and 9500);
```

409. 우리반 테이블의 성별 컬럼에 '남 아니면 여' 만 입력되게끔 CHECK제약을 거시오!

```
alter table emp12
   add constraint emp12_gender_ck check( gender in ('남','여'));
```

410. 다음 결과에서 짝수만 출력하시오!

보기 :

```
with test_table as (select level as num1
                           from dual
                           connect by level <=10)
select num1
   from test_table;
```

정답 :

```
with test_table as (select level as num1
                           from dual
                           connect by level <=10)
select num1
   from test_table
   where mod(num1,2) = 0;
```

411. with절로 구구단 2단을 출력하시오.

```
with test_table as (select level as num1
                           from dual
                           connect by level <=9)
select '2 x '||num1||'= '||num1*2
   from test_table;
```

※ 설명 : 오라클은 크게 두가지로 나뉘는데 메모리(단어장)와 디스크로(사전) 나뉜다. as 다음에 괄호 안에 나오는 쿼리문의 결과가 하드 디스크에 저장이 되고 테이블 명이 test_table이 된다.
인라인뷰로 하면 메모리에 저장되고 with 절로 쓰면 디스크에 저장된다.

412. with 절로 구구단 전체를 출력하시오.

```
2 X 1 = 2
2 X 2 = 4
.
.
.
9 X 9 = 81
```

문제 365번의 답 갖고와서 풀기 (from 절의 서브쿼리)

```
select a.num1||'x'||b.num2||'='||a.num1*b.num2
```

```
from (select level num1
        from dual
        where level !=1
        connect by level <=9)a,
        (select level num2
            from dual
            connect by level <=9) b;
```

위의 인라인뷰 SQL문을 with절로 변경합니다.

```
with temp_table1 as (select level num1
        from dual
        where level !=1
        connect by level <=9),
     temp_table2 as (select level num2
                       from dual
                       connect by level <=9)
select a.num1||'x'||b.num2||'='||a.num1*b.num2
     from temp_table1 a, temp_table2 b;
```

또는

```
with temp_table1 as (select level num1
                  from dual
                  connect by level <=9),
            temp_table2 as (select level num2
                              from dual
                              connect by level <=9)
     select a.num1||'x'||b.num2||'='||a.num1*b.num2
from temp_table1 a, temp_table2 b
where a.num1 between 2 and 9;
```

413. 아래의 인라인뷰 SQL을 with절로 변경하시오!

```
select e.ename, e.sal, e.deptno, v.부서평균
     from emp e, (select deptno, avg(sal) 부서평균
                  from emp
                  group by deptno) v

     where e.deptno = v.deptno and e.sal > v.부서평균
```

※ 정답 :

```
with temp_table as (select deptno, avg(sal) 부서평균
                    from emp
          group by deptno)
select e.ename, e.sal, e.deptno, v.부서평균
     from emp e, temp_table v
 where e.deptno = v.deptno and e.sal > v.부서평균;
```

※ 설명 : as 다음에 나오는 쿼리문의 결과가 아주 많거나 또는 이 쿼리문에 인라인뷰로 쓰였을때는 여러 번 작성될 경우 with절이 유용

414. (SQL 알고리즘 15번) with절을 이용해서 직각 삼각형을 출력하시오!

★

★★

★★★
★★★★
★★★★★

```
with temp_table as (select level as num1
                    from dual
                    connect by level <=9)

select ....?

with temp_table as (select level as num1
                    from dual
                    connect by level <=9)
select lpad('★', num1, '★')
   from temp_table;
```

415. 이번에는 숫자를 물어보게 하고 숫자를 입력하면 해당 숫자만큼 직각 삼각형이 만들어지게 하시오!

```
with temp_table as (select level as num1
                    from dual
                    connect by level <= &숫자)
select lpad('★', num1, '★')
   from temp_table;
```

416. (SQL 알고리즘 16번) 이번에는 삼각형이 출력되게 하시오!

숫자를 입력하세요 ~ 5

```
    ★
   ★★
  ★★★
 ★★★★

with temp_table as (select level as num1
                    from dual
                    connect by level <= &숫자)
select rpad(' ', &숫자-num1)||lpad('★', num1,'★')
   from temp_table;
```

417. 직업, 직업별 토탈월급을 출력하시오! (세로 출력)

```
select job, sum(sal)
      from emp
      group by job;
```

418. 위의 직업별 토탈월급들 중에서의 평균값을 출력하시오

```
select avg(sum(sal))
   from emp
   group by job;
```

419. 문제 417번 쿼리문과 문제 418번 쿼리문을 둘다 이용해서 직업, 직업별 토탈월급을 출력하는데 직업별 토탈월급이 직업별 토탈월급들의 평균값보다 더 큰것만 출력하시오.

```
select job, sum(sal)
   from emp
   group by job
```

```
having sum(sal) > (select avg(sum(sal))
                   from emp
                   group by job);
```

- with 절의 장점? 똑같은 SQL을 하나의 SQL내에서 여러 개 사용될 때 유용합니다.

420. 위의 SQL (419번 문제)을 with절로 구현하시오!

```
with job_sumsal as (select job, sum(sal) as 토탈
                    from emp
                    group by job)
select job, 토탈
   from job_sumsal
   where 토탈 > (select avg(토탈)
                 from job_sumsal);
```

with절 내에서 그룹함수를 쓰게되면 그룹함수에 컬럼별칭을 주어야 합니다.

※ 설명 : 복잡한 쿼리내에 동일 쿼리블럭이 두번이상 발생하는 경우에 사용하면 좋은 성능을 보이는 SQL이 with절 입니다.

421. 위의 SQL의 실행계획을 확인해서 하드디스크에 TEMP 테이블로 만들어지는지 확인하시오!

```
explain plan for
with job_sumsal as (select job, sum(sal) as 토탈
                    from emp
                    group by job)
select job, 토탈
   from job_sumsal
   where 토탈 > (select avg(토탈)
                 from job_sumsal);
 select * from table(dbms_xplan.display);
```

```
| Id  | Operation                          | Name                 | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                   |                      |    14 |   266 |     8  (13)| 00:00:01 |
|   1 |  TEMP TABLE TRANSFORMATION         |                      |       |       |            |          |
|   2 |   LOAD AS SELECT (CURSOR DURATION MEMORY)| SYS_TEMP_0FD9D6648_1FCBF8 |       |       |            |          |
|   3 |    HASH GROUP BY                   |                      |    14 |   266 |     4  (25)| 00:00:01 |
|   4 |     TABLE ACCESS FULL              | EMP                  |    14 |   266 |     3   (0)| 00:00:01 |
|*  5 |   VIEW                             |                      |    14 |   266 |     2   (0)| 00:00:01 |
|   6 |    TABLE ACCESS FULL               | SYS_TEMP_0FD9D6648_1FCBF8 |    14 |   266 |     2   (0)| 00:00:01 |
|   7 |    SORT AGGREGATE                  |                      |     1 |    13 |            |          |
|   8 |     VIEW                           |                      |    14 |   182 |     2   (0)| 00:00:01 |
|   9 |      TABLE ACCESS FULL             | SYS_TEMP_0FD9D6648_1FCBF8 |    14 |   266 |     2   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------------------------
```

TEMP TABLE이 작성되었음을 알 수 있다.

422. 직각삼각형을 출력하는 415번 문제를 다시 푸는데 아래와 같이 입력 메세지창에서 나오는 메세지가 숫자를 입력하세요~ 라고 나오게하시오~

보기 ---

```
accept p_num1 prompt '숫자를 입력하세요~'
with temp_table as (select level as num1
                    from dual
                    connect by level <=&p_num1)
select lpad('★', num1, '★')
   from temp_table;
```

답변 :

```
accept p_num1 prompt '숫자를 입력하세요~'
with temp_table as (select level as num1
         from dual
         connect by level <=&p_num1)
select lpad('★', num1, '★')
       from temp_table;
```

423. (SQL알고리즘 17번) 아래와 같이 숫자를 물어보게하고 숫자를 입력하면 마름모가 출력되게 하시오.

숫자를 입력하세요 ~ 5

```
      ★
     ★★
    ★★★
   ★★★★
  ★★★★★
   ★★★★
    ★★★
     ★★
      ★
```

```
with temp_table as (select level as num1
                    from dual
                    connect by level <= &&숫자)
select rpad(' ', &숫자-num1)||lpad('★', num1,'★')
   from temp_table
union all
select rpad('★', &숫자-num1)||lpad(' ', num1,' ')
   from temp_table;
```

(with절 X)

```
select rpad(' ', 5-level)||lpad('★', level,'★')
          from dual
        connect by level <=5
union all
select rpad(' ', level)||lpad('★', 5-level, '★')
   from dual
   connect by level <= 4;
```

(with절 O)

```
accept p_num1 prompt '숫자를 입력하세요~'
with temp_table as (select level as num1
                        from dual
                        connect by level <= 2*&p_num1)
select lpad(' ', &p_num1 - num1, ' ') || lpad('★', num1, '★') 마름모
   from temp_table
   where num1 <= &p_num1
union all
select lpad(' ', num1 - &p_num1, ' ') || lpad('★', (2*&p_num1) - num1, '★') 마름모
```

```
from temp_table
where num1 > &p_num1;
```

424. (SQL알고리즘 18번) 아래와 같이 두 숫자를 각각 입력받아 사각형을 출력하시오!

가로의 숫자를 입력하세요 ~ 5

세로의 숫자를 입력하세요~ 4

힌트 : accept 두번 쓰기

```
accept p_num1 prompt '가로의 숫자를 입력하시오'
accept p_num2 prompt '세로의 숫자를 입력하시오'
```

★★★★★

★★★★★

★★★★★

★★★★★

```
accept p_num1 prompt '가로의 숫자를 입력하시오'
accept p_num2 prompt '세로의 숫자를 입력하시오'
select lpad('★',  &p_num1, '★')
    from dual
    connect by level <= &p_num2;
```

425. (SQL 알고리즘 19번) SQL로 최대공약수를 구하시오.

첫번째 숫자를 입력하세요 ~ 16

두번째 숫자를 입력하세요 ~ 24

최대 공약수는 8입니다.

힌트 : level mod (16, level) mod(24,level)

(with절 X)

```
accept p_num1 prompt '첫번째 숫자를 입력하세요~'
accept p_num2 prompt '두번째 숫자를 입력하세요~'
select '최대 공약수는 '||max(level)||'입니다.'
    from dual
    where mod(&p_num1,level)=0 and mod(&p_num2,level)=0
    connect by level <= 10;
```

(with절 O)

```
accept p_num1 prompt '첫번째 숫자를 입력하세요~'
accept p_num2 prompt '두번째 숫자를 입력하세요~'
explain plan for
with temp_table as (select level as num1
                    from dual
                    connect by level <= &p_num2)
select '최대 공약수는 '||max(num1)||'입니다.'
    from temp_table
```

```
where mod(&p_num1,num1)=0 and mod(&p_num2,num1)=0;
select * from table(dbms_xplan.display);
```

426. 이름과 부서위치와 월급과 부서번호를 출력하시오.
(emp와 dept를 조인하세요)

```
select e.ename, d.loc, e.sal, e.deptno
    from emp e, dept d
    where e.deptno = d.deptno;
```

427. emp와 dept와 salgrade테이블을 조인해서 이름과 월급과 부서위치와 급여등급(grade)을 출력하시오!

```
select e.ename, e.sal, d.loc, s.grade
    from emp e, dept d, salgrade s
    where e.deptno = d.deptno and e.sal between s.losal and s.hisal;
```

428. 아래와 같이 bonus라는 테이블을 생성하시오!

```
create table bonus
as
select empno, sal*1.5 as comm2
      from emp;
```

429. 사원이름, 월급, 부서위치, comm2를 출력하시오!

```
select e.ename, e.sal, d.loc, b.comm2
    from emp e, dept d, bonus b
    where e.deptno = d.deptno and e.empno = b.empno;
```

430. 위의 결과에서 등급이 2000이상인 사원들을 출력하시오.

```
select e.ename, e.sal, d.loc, b.comm2
    from emp e, dept d, bonus b
    where e.deptno = d.deptno and e.empno = b.empno and e.sal >= 2000;
```

431.

정답:

```
accept p_num1 prompt '밑변을 입력하세요~'
accept p_num2 prompt '높이를 입력하세요~'
accept p_num3 prompt '빗변을 입력하세요~'
```

```
select case when power(&p_num1,2) + power(&p_num2,2) = power(&p_num3,2) then '직각 삼각형이 맞습니다.'
        else '직각삼각형이 아닙니다.' end as "피타고라스의 정리"
        from dual;
```

432. ext_emp11 테이블을 조회하여 직업이 SALESMAN인 사원들의 이름과 직업과 월급을 출력하시오!

```
select ename, job, sal
from ext_emp11
where job = 'SALESMAN';
```

433. ext_emp11과 dept를 조인해서 이름과 부서위치를 출력하시오!

```
select e.ename, d.loc
    from ext_emp11 e, dept d
    where e.deptno = d.deptno;
```

434. ext_emp11의 scott의 월급을 6000으로 갱신하시오!

```
update table ext_emp11
    set sal = 6000
    where ename = 'SCOTT';
```

435. dept.txt를 외부테이블로 생성해서 아래와 같이 select할 수 있도록 하시오!

```
select *
    from ext_dept
```

1. dept.txt를 카페에서 내려받고 c드라이브 밑에 data폴더 밑에 둔다.
2. ext_emp11테이블 스크립트를 가져와서 exp_dept테이블을 만들 수 있도록 수정하시오!

```
create table ext_dept
    ( EMPNO    NUMBER(10),
      ENAME    VARCHAR2(20),
      JOB      VARCHAR2(20),
      MGR      NUMBER(10),
      HIREDATE  DATE,
      SAL    NUMBER(10),
      COMM    NUMBER(10),
      DEPTNO   NUMBER(10))
      organization external
     (type oracle_loader
      default directory emp_dir
      access parameters
      (records delimited by newline
       fields terminated by ","
      (empno char,
       ename char,
       job char,
       mgr char,
       hiredate date "yyyy/mm/dd",
       sal char,
       comm char ,
       deptno  char  ) )
       location ('emp.txt') );
```

436. 이름에 EN또는 IN을 포함하고 있는 사원들의 이름과 월급을 출력하시.

```
select ename, sal
```

```
from emp
where ename like '%EN%' or ename like '%IN%';
```

437. 문제 436의 SQL을 정규 표현식 함수인 regexp.like를 이용해서 출력하시오!

- 정규 표현식 함수

1. regexp_substr
2. regexp_like
3. regexp_count
4. regexp_instr
5. regexp_replace

정답:

```
select ename, sal
      from emp
      where regexp_like(ename, 'EN|IN');
```

438. 우리반 테이블에서 전공이 통계, 수학, 컴퓨터, 전자가 포함된 학생들의 이름과 전공을 출력하시오!

```
select ename, major
   from emp12
   where major like '%통계%' or major like '%수학%' or major like '%컴퓨터%' or major like '%전자%';

select ename, major
   from emp12
   where regexp_like(major, '통계|수학|컴퓨터|전자');
```

439. employees.csv 파일을 오라클의 테이블로 생성하시오.

```
create   table  employees
( EMPLOYEE_ID         number(10),
 FIRST_NAME           varchar2(20),
 LAST_NAME            varchar2(20),
 EMAIL                varchar2(50),
 PHONE_NUMBER     varchar2(30),
 HIRE_DATE            date,
 JOB_ID               varchar2(20),
 SALARY               number(10,2),
 COMMISSION_PCT    number(10,2),
 MANAGER_ID           number(10),
 DEPARTMENT_ID       number(10) );
```

employees,csv파일을 sqldeveloper를 이용해서 employees 테이블에 로드한다.

데이터 임포트 하기

440. 이름의 첫글자가 st로 시작하면서 끝글자가 en으로 끝나는 사원들의 first_name을 출력하시오!

```
select first_name
   from employees
   where substr(first_name, 1,2) = 'St' and first_name like '%en';

select first_name
      from employees
      where regexp_like(first_name, '^St(.)+en$');
```

❖ 설명 : ^는 시작을 나타낸다. $는 끝을 나타낸다.
시작이 st로 시작하면서 끝이 en으로 끝나는 first_name을 찾는다.
가운데 있는 "(.)+"은 점(.)을 한자리를 나타내는데 +가 여러개를 나타내는 것이므로 한자리가 여러개인것을 나타낸다. St와 en사이에 여러개의 철자가 와도 된다는 뜻입니다.

441. 우리반 테이블에서 성씨가 김씨 또는 허씨이고 끝의 글자가 '민'인 학생들의 이름을 출력하시오!

```
select ename
      from emp12
   where regexp_like(ename, '^(김|허)(.)+민$');
```

442. 겨울왕국(winter_kingdom)에는 elsa가 몇번 나오냐?

```
select win_text, regexp_count(lower(win_text), 'elsa') as 건수
      from winter_kingdom;
---------------------------------------------------------------
전체건수 구하기
select sum(건수)
      from(select win_text, regexp_count(lower(win_text), 'elsa') as 건수
from winter_kingdom);
```

443. 우리반 테이블에서 이름과 나이를 출력하는데 나이를 출력할 때 아래와 같이 앞의 숫자를 *로 출력되게 하시오!

이준혁 *9
한 *1
현지연 *5

```
select ename, regexp_replace(substr(age,1,1), '[0-9]', '*')||substr(age,2,1) as 나이
from emp12;
```

444. 우리반 테이블(emp12)를 emp12_a, emp12_b, emp12_c로 구조만 가져와서 만들고 무조건 all insert문으로 emp12테이블의 전체 데이터 3개를 테이블엡 입력하시오.

```
create table emp12_a
   as
   select *
      from emp12
      where 1=2;

create table emp12_b
   as
   select *
      from emp12
      where 1=2;

create table emp12_c
   as
   select *
      from emp12
      where 1=2;

insert all into emp12_a
            into emp12_b
            into emp12_c
```

```
select *
    from emp12;
```

445. emp12 테이블의 데이터를 아래의 3개의 테이블에 입력하는데 통신사가 sk면 emp12_sk에 입력하고 lg면 emp12_lg에 입력하고 kt면 emp12_kt에 입력하시오!

```
create table emp12_sk as select *from emp12 where 1=2;
create table emp12_lg as select *from emp12 where 1=2;
create table emp12_kt as select *from emp12 where 1=2;

insert all
    when telecom='sk' then into emp12_sk
    when telecom='lg' then into emp12_lg
    when telecom='kt' then into emp12_kt
select ename, age, gender, major, telecom, email, address, telecom_price
    from emp12;
```

***번외* (SQL알고리즘 21번) 몬테카를로 알고리즘을 원주율을 구하시오.**

```
accept p_num1 prompt '정사각형안 총 점의 개수를 입력하세요'
with temp_table as (select /*+ materialize */dbms_random.value(0,1) x, dbms_random.value(0,1) y
                    from dual
                    connect by level <=&p_num1)
select 4*count(*)/&p_num1 as 파이값
   from temp_table
   where power(x,2)+ power(y,2) <=1;
```

446. week_sum_sal 에서 이름, 일당을 다 합친 주급을 출력하시오!

```
select ename, mon+tues+wed+thurs+fri
    from week_sum_sal;
```

※ 설명 : 위와같이 sum 그룹함수를 이용하지 못하고 컬럼의 데이터를 일일이 더해서 SQL문장을 작성하면 번거로운 코딩이 됩니다.

- pivoting insert문으로 구현

```
create table sales_info
(ename    varchar2(10),
sal          number(10) );

insert all into sales_info values(ename, mon)
            into sales_info values(ename, tues)
            into sales_info values(ename, wed)
            into sales_info values(ename, thur)
            into sales_info values(ename, fri)
select ename, mon, tues, wed, thur, fri
    from week_sum_sal;
```

447. (11/20 시험문제 유형1) 직업, 직업별 토탈월급을 출력하는데 직업이 SALESMAN은 제외하고 출력하고 직업별 토탈월급이 4000이상인 것만 출력하고 직업별 토탈월급이 높은 것부터 출력하시오.

```
select job, sum(sal)
   from emp
   where job != 'SALESMAN'
   group by job
```

```
having sum(sal) >=4000
order by sum(sal) desc;
```

448. (11/20 시험문제 유형2) 이름, 직업, 월급을 출력하는데 직업은 abcd순서대로 출력하고 직업을 abcd순서대로 출력한 것을 기준으로 월급이 높은것부터 출력하시오!

```
select ename, job, sal
    from emp
    order by job asc, sal desc;
      (order by 2 asc, 3 desc)라고도 할 수 있음
```

콤마써서 여러개를 이을 수 있음.

449. (11/20 시험문제 유형2 연습) 이름, 부서번호, 입사일을 출력하는데 부서번호를 ascending하게 출력하고 부서번호가 asceding하게 출력된 것을 기준으로 입사일을 descending하게 출력하시오!

```
select ename, deptno, hiredate
    from emp
    order by deptno asc, hiredate desc;
```

450. (11/20 시험문제 유형3) 직업, 직업별 토탈월급을 출력하는데 맨 아래에 전체 토탈 월급이 출력되게 하시오.

```
select job, sum(sal)
    from emp
    group by rollup(job);
```

451. (11/20 시험문제 유형3 연습) 위의 문제를 grouping sets로 출력하시오.

```
select job, sum(sal)
    from emp
    group by grouping sets((job), ());
```

452. (11/20 시험문제 유형3 연습) 아래 SQL의 결과를 union all 로 변경하시오.

```
select deptno, job, sum(sal)
      from emp
      group by grouping sets((deptno), (job));
```

(rollup과 cube와 grouping sets와 같은 레포팅함수 쓰지 말고 순수하게 union all로만 구현하세요.)

```
select to_number(null) as deptno, job, sum(sal)
    from emp
    group by job
union all
select deptno, to_char(null) as job, sum(sal)
    from emp
    group by deptno;
```

※ 설명: union all 위아래의 쿼리의 컬럼의 갯수가 동일해야 합니다.
union all 위아래의 쿼리의 컬럼의데이터 타입이 동일해야 합니다.
union all 위아래의 쿼리의 컬럼의 이름도 동일해야 order by 절 사용할 때 확실하게 정렬이 됩니다.

453. (11/20 시험문제 유형3 연습) 아래의 SQL을 union all로 변경하시오!

```
select deptno, job, sum(sal)
      from emp
```

group by grouping sets ((deptno), job, ());

정답:

```
select to_number(null) as deptno, job, sum(sal)
   from emp
   group by job
union all
select deptno, to_char(null) as job, sum(sal)
   from emp
   group by rollup(deptno)
union
select to_number(null) as deptno, to_char(null) as job, sum(sal)
   from emp;
```

454. (11/20 시험문제 유형4) 부서위치, 부서위치별 토탈월급을 출력하는데 부서위치별 토탈월급을 출력할 때 천단위 콤마를 부여해서 출력하시오!

```
select d.loc, to_char(sum(e.sal), '999,999')
   from emp e, dept d
   where e.deptno = d.deptno
   group by d.loc;
```

455. (11/20 시험문제 유형5) 부서위치, 부서위치별 토탈월급을 출력하는데 가로로 출력하시오!

```
NEW YORK      DALLAS     CHICAGO
8750           10875        9400

select  sum( decode( d.loc, 'NEW YORK', e.sal ) )   as "NEW YORK",
        sum( decode( d.loc, 'DALLAS',  e.sal)   ) as   "DALLAS",
        sum( decode( d.loc, 'CHICAGO', e.sal)   ) as  "CHICAGO"
 from  emp e, dept  d
 where  e.deptno = d.deptno;
```

456. (11/20 시험문제 유형6) ALLEN보다 더 늦게입사한 사원들의 이름과 입사일을 출력하는데 최근에 입사한 사원부터 출력하시오!

```
select ename, hiredate
   from emp
   where hiredate > (select hiredate
                     from emp
                     where ename = 'ALLEN')
   order by hiredate desc;
```

457. (11/20 시험문제 유형7) 관리자가 아닌 사원들의 이름을 출력하시오!
(자기 밑에 직속부하가 한명도 없는 사원들을 출력하시오)

```
select ename
   from emp
   where empno not in (select mgr
                       from emp
                       where mgr is not null);
```

458. 직업, 이름, 월급, 순위를 출력하는데 순위가 직업별로 각각 월급이 높은순서대로 순위를 부여하시오.

```
select job, ename, sal, dense_rank() over (partition by job
                                  order by sal desc) 순위
   from emp;
```

459. (11/20 시험문제 유형8) 문제 458번의 결과를 다시 출력하는데 순위가 1등인 사원들만 출력하시오!

```
select *
   from (select job, ename, sal, dense_rank() over (partition by job
                                                    order by sal desc) 순위
                                    from emp)
   where 순위 = 1;
```

460. (11/20 시험문제 유형9) 아래의 테이블을 생성하고 empno에 primary key제약을 거시오.

테이블 명 : emp460

컬럼명 : empno <--- primary key제약을 거세요~
ename
sal
hiredate

```
create table emp460
(empno number(10),
       ename varchar(10),
       sal  number(10),
       hiredate date);

alter table emp460
      add constraint emp460_empno_pk primary key(empno);
```

461. (11/20 시험문제 유형9 연습) 아래의 데이터를 담는 테이블을 emp461로 생성하고 아래의 데이터를 력하고 ename에 unique제약을 거시오.

empno	ename	sal	hiredate
1111	scott	3000	81/11/17
2222	smith	4000	82/12/21
3333	allen	5000	83/05/02

```
create table emp461
   (empno  number(10),
   ename   varchar2(10),
   sal     number(10),
   hiredate date);

insert into emp461
   values (1111, 'scott', 3000, to_date('81/11/17', 'RR/MM/DD'));
   insert into emp461
   values (2222, 'smith', 4000, to_date('82/12/21', 'RR/MM/DD'));
    insert into emp461
   values (2222, 'allen', 5000, to_date('83/05/02', 'RR/MM/DD'));

alter table emp461
   add constraint emp416_ename_un unique(ename);
```

462. (11/20 시험문제 유형10) 이름, 월급, 부서번호, 자기가 속한 부서번호의 평균월급을 출력하시오!

```
select e.ename, e.sal, e.deptno, v.부서평균
   from emp e, (select deptno, avg(sal) 부서평균
         from emp
         group by deptno) v
```

```
    where e.deptno= v.deptno;
```

463. (11/20 시험문제 유형10 연습)위의 SQL을 with절로 구현하시오

```
with temp_table1 as
(select deptno, avg(sal)부서평균
    from emp
    group by deptno)
select e.ename, e.sal, e.deptno, t.부서평균
    from emp e, temp_table1 t
    where t.deptno = e.deptno;
```

464. (11/20 시험문제 유형11) SQL알고리즘/ 1부터 10까지의 숫자중에 홀수만 출력하시오.

```
select level
    from dual
    where mod(level,2) = 1
    connect by level <=10;
```

또는

```
with num_table as (select level as num1
                        from dual
                        connect by level <=10)
select num1
      from num_table
      where mod(num1,2) = 1;
```

465. allen이라는 유저를 패스워드 tiger1234로 생성하고 allen으로 접속할 수 있도록 접속 권한을 부여하고 allen으로 접속해 보시오.

```
create user allen
    identified by tiger1234;

grant connect to allen;
```

도스창에 sqlplus allen/tiger1234 라고 친다.
or connect allen/tiger1234

466. allen 유저에서 아래의 테이블을 생성하시오.

테이블 명 : emp466
컬럼명 : empno, ename, sal

```
create table emp466
   (empno number(10),
    ename varchar2(10),
     sal number(10));
```

이렇게 하면

```
---------------------------------
```

1행에 오류:
ORA-01031: 권한이 불충분합니다
라고 나온다.

- create table권한을 allen 유저에게 부여하는 방법:

sys 유저에서 grant create table to allen;
(sys 유저는 dba유저라서 테이블 생성할 수 있는 권한을 줄 수 있다)

467. king 유져도 삭제하시오!

SQL> drop user king cascade;